<http://www.diva-portal.org>

Postprint

This is the accepted version of a paper published in *Journal of Systems and Software*. This paper has been peer-reviewed but does not include the final publisher proof-corrections or journal pagination.

Citation for the original published paper (version of record):

Asplund, F., Törngren, M. (2015)

The Discourse on Tool Integration Beyond Technology, A Literature Survey.

Journal of Systems and Software, 106: 117-131

Access to the published version may require subscription.

N.B. When citing this work, cite the original published paper.

Permanent link to this version:

<http://urn.kb.se/resolve?urn=urn:nbn:se:kth:diva-169635>

The Discourse on Tool Integration Beyond Technology, A Literature Survey

Fredrik Asplund^{a,1,*}, Martin Törngren^a

^a*KTH Royal Institute of Technology, Department of Machine Design, Division of Mechatronics, Brinellvägen 83, 10044 Stockholm, Sweden*

Abstract

The tool integration research area emerged in the 1980s. This survey focuses on those strands of tool integration research that discuss issues beyond technology.

We reveal a discourse centered around six frequently mentioned non-functional properties. These properties have been discussed in relation to technology and high level issues. However, while technical details have been covered, high level issues and, by extension, the contexts in which tool integration can be found, are treated indifferently. We conclude that this indifference needs to be challenged, and research on a larger set of stakeholders and contexts initiated.

An inventory of the use of classification schemes underlines the difficulty of evolving the classical classification scheme published by Wasserman. Two frequently mentioned redefinitions are highlighted to facilitate their wider use.

A closer look at the limited number of research methods and the poor attention to research design indicates a need for a changed set of research methods. We propose more *critical* case studies and method diversification through theory triangulation.

Additionally, among disparate discourses we highlight several focusing on standardization which are likely to contain relevant findings. This suggests that open communities employed in the context of (pre-)standardization could be especially important in furthering the targeted discourse.

Keywords: Tool Integration, Support Environments

1. Introduction

Tool integration is a cross-disciplinary research area incorporating influences from many fields, such as *Software Engineering*, *Systems Engineering*, *Human-Machine Interaction* and *Economics*. Buxton's STONEMAN report is often

*Corresponding author

Email addresses: fasplund@kth.se (Fredrik Asplund), martint@kth.se (Martin Törngren)

¹Phone: +46 8 790 7405, Fax: +46 8 20 22 87

mentioned as a starting point for the discussion on tool integration (Buxton, 1980). Buxton (1980) specified the requirements for a *support environment* for programming Ada by defining the appropriate tools, tool integration mechanisms and interfaces, but also introduced the notion of integrating tools throughout a software project life-cycle. During the 1980s a plethora of initiatives to specify support environments followed, the most well known being the European Portable Common Tool Environment (PCTE) initiative. In the late 1980s and early 1990s, this carried over into an intense academic discussion regarding many different types of support environments. It was already clear at this point that the research on tool integration consisted of several different strands of research (Brown, 1993a). The identified strands currently include the (overlapping) categories of tool integration versus mechanisms, technology, frameworks, semantics, modelling, process, dimensions, types, standards and industrial experience (Brown, 1993a; Wicks, 2004; Maalej, 2009). Throughout the last two decades, the strand that has seen the majority of the activity is the one that focuses on the *technology*, i.e. the separate mechanisms for achieving tool integration (Wicks and Dewar, 2007). Many valuable findings and insightful discussions are found in this particular strand of research, for instance those related to technological innovations such as Eclipse and Open Services for Lifecycle Collaboration (OSLC). The former is an innovative plug-in framework technology that once turned the entire tool integration market upside down, while the latter is a web API technology that currently shows promise of a large impact. However, the other strands of work are also important, although their influence is currently much more difficult to appraise.

This paper focuses on those strands of tool integration research *that have implications beyond a specific technology*. It contributes to the body of knowledge in tool integration by providing an exploratory literature survey focused on the issues with (and discussion of) tool integration that go beyond solving technological challenges. Our hope is that this will support disruptive change. Incremental changes to technology are valuable, but progress in the tool integration field has been painstakingly slow. If the solutions provided by academia cannot gain traction and impact within industry, then our understanding of industry must be flawed. Identifying missing knowledge might eventually facilitate more relevant technological choices. It could also lead to the removal of unknown, non-technological obstacles hindering the successful deployment of tool integration. Furthermore, it should point to changes to current research approaches to allow for more efficient, conclusive research into the field.

The background is our part in the iFEST project (iFEST Consortium, 2013), an EU research project focusing on the specification and implementation of an integration framework for establishing and maintaining tool chains to engineer complex industrial embedded systems. While building support environments is a challenging task due to the sheer complexity of today's technology, many of the difficulties encountered during iFEST were not linked to technology *per se*. The choice of a particular approach or technology could make perfect sense to one stakeholder, while another discounted it outright. The ensuing discussions pointed at a lack of adequate research into more high level questions, such as how

to prioritize between business models, stakeholders or even different academic discourses.

To avoid a situation in which discussions would have degenerated into a mere battle of wills, and to enable an unbiased approach to tool integration, we chose early on to focus on the strands of research that try to reach an *overall* understanding of what tool integration *is*. Thus, by identifying the essential core of the cross-disciplinary discourse related to tool integration, we aimed to facilitate future decisions on tradeoffs and identify any weaknesses in the discourse that may make such decisions difficult. To achieve this, we designed a literature survey that focused on what we called the *essence of tool integration* - how it is discussed, the context of this discussion and what the implications are. In other words, the survey focused on the non-technical aspects of the tool integration literature, such as how tool integration is defined, if the concept can be further divided into separate parts, what its purpose is and what is required to achieve it. In addition, the survey considered when these types of questions tended to arise and to what purpose. This also means that we have tried to go beyond discussing such things as individual meta-models, reference models and patterns, at least beyond what is motivated by our approach. While these capture important aspects of tool integration at an abstract level, they focus on functionality and usually do not cover the even higher levels of abstraction targeted by this study.

The basis for the paper is, as will be explained in the subsequent sections, a paper by Wasserman (1990). This paper is a widely recognized seminal paper in the strands of research focusing on issues of tool integration beyond technology. The status of this paper stems from its definition of what later became a much used classification scheme based on different “dimensions” of tool integration, namely *Control*, *Data*, *Platform*, *Presentation* and *Process Integration*. It has been popular to use these dimensions as support when reasoning about tool integration. This scheme is further described in Section 5.

The paper is divided into five distinct parts due to the exploratory nature of the study. The first part defines the questions that guided our exploratory investigation (Section 1) and motivates the approach towards answering these questions (Section 2). The second part discusses how these questions led to the allocation of the surveyed papers into initial categories based on common traits or unique contributions related to the initial questions (Section 3). In the third part these categories are used to elicit and analyse four ways in which the discussion of tool integration that go beyond solving technological challenges is either strong or weak (separate discussions in Sections 4 to 7). Which conclusions that are possible to draw based on these analyses is discussed in the fourth part of the paper (Sections 8). Finally, the core findings and conclusions are summarized in the last part (Section 9).

2. An Iterative Literature Survey

This section starts with explaining the approach of this literature survey. A case is then made for the validity of the research findings based on the approach

and extra precautions taken.

2.1. The Approach

The findings presented in this paper come from an iterative literature survey, which took place over a period of four years.

The first iteration, in which the *State of the Art* of tool integration was studied, took place early 2010 during the start of the iFEST project. The 39 sources studied during this iteration consisted of most² of the *Association for Computing Machinery* (ACM) Digital Library (Association for Computing Machinery, 2013) database citation list for Wasserman (1990).

The second iteration took place between 2010 and 2012, at the same time as the main part of the iFEST project. Made up of a consortium of 21 partners, consisting of international companies and universities, much input was obtained on different approaches to tool integration. When compiling the most interesting work obtained in regard to the essence of tool integration, it became obvious that most of these sources were based on or oriented around Wasserman (1990).

The third iteration took part from late 2012 to early 2013 and focused on the sources in the ACM Digital Library (Association for Computing Machinery, 2013) and the Google Scholar (Google, 2013) databases which cite Wasserman (1990). All highly cited sources³ from 1990 to early 2013 were included. Furthermore, *all* sources issued from 2007 to 2012 were included in the study regardless of how many times they had been cited. These criteria ensured that all relevant sources received from iFEST partners during the previous iteration were formally included. At this time a total of 75 relevant sources had been identified during the second and third iteration⁴. Based on the discussion in these sources, a further 15 sources of interest were identified, bringing the total number surveyed during the second and third iteration up to 90. In practice this primarily involved using citations to backtrack to sources discussing classification schemes other than Wasserman's.

In the final iteration the whole set of sources were surveyed again to summarize and double-check the data presented in this paper. Out of 129 sources, 117 were eventually used as a basis for the survey. The 12 sources excluded were deemed not to contribute to the targeted discussion, i.e. the discussion of what we called the *essence of tool integration*. This decision was based on a careful reading of the complete sources after which we could not include them in any of the categories described in Section 3. This does not reflect on the quality of these sources or their usefulness in surveys with other focuses.

²One source was not possible to obtain.

³A high count was defined as 20 citations or more.

⁴In comparison with the first iteration, the citation lists contained 90 additional sources fitting the criteria at this time. However, 1 source did actually not refer to Wasserman (1990), 7 sources could not be used due to language difficulties and 7 sources proved to be inaccessible. Only 4 of the excluded sources were from the highly cited category.

2.2. The Validity of the Findings

There is much advice to be found on how to conduct a literature survey, but time must be spent on *research design* to ensure the trustworthiness of the findings. The approach described in the previous subsection is primarily the result of considering the implications of the survey’s setup on *validity*.

Experimental researchers commonly divide validity into internal and external validity, i.e. the degree to which a study can “*measure what it aims to measure*”, and the degree to which the research findings are generalizable to other “*populations, settings, treatment variables and measurement variables*” respectively (Kothari, 2004). This leads to a strong emphasis on validity, with literature studies often used as one way of proving the internal validity of an experimental study. Other researchers have challenged both these definitions of validity and the priority given to ensuring it, for example Glaser and Strauss in their discussion on the generation of theory through Grounded Theory (Glaser and Strauss, 1967). Nevertheless, validity remains an important issue to consider in any given research design.

Given the aim of this literature survey (to understand how the essence of tool integration is discussed, the context of this discussion and what the implications are) the issue of external validity has been treated as a coverage problem. Instead of claiming the possibility to generalize to other sources and focusing on sampling, the scope of the literature survey was set early on to all sources that cited Wasserman (1990). Two consequences of this approach ensured a high coverage of sources related to the discourse on the essence of tool integration: Firstly, most papers discussing the essence of tool integration somehow orient themselves in regard to the most seminal papers in the field, of which Wasserman (1990) is one; and secondly, any high quality source that does not cite Wasserman (1990) is likely to be cited by at least some of the subsequently published sources on the essence of tool integration. The latter means that high quality sources missed by the early iterations could be identified by backtracking from sources in later iterations. Two additional choices were also made to eliminate potential sources of selection bias: the choice of databases included both a traditional research database (Association for Computing Machinery, 2013) and a Big Data research database (Google, 2013); and, even though high citation rankings were used to identify relevant historical sources, the sources from 2007 to 2012 were included regardless of their citation rank. This ensured that sources that have not been around for long enough to be widely recognized were included anyway.

We can thus be reasonably confident that we have captured the *relevant* parts of the *targeted* discourse (particular strands of research in the tool integration literature). However, discourses are dynamic things, which require a sharing of authors, concepts and terminology to connect. There may well be separate discourses that contain relevant findings, but which researchers active in the discourse on tool integration either never encounter or immediately dismiss because the relevance is not obvious. When researchers are able to establish an unexpected connection between two discourses previously thought of as disparate, the results may however be far-reaching. Consider for instance when

Reynolds (1987) introduced a computer model mimicking animal aggregation. This work, based on input from biology, has eventually inspired research in such distant fields as networked control and robot navigation. Unfortunately it is not trivial to establish connections between disparate, but principally related, discourses. Therefore, we choose to sample our sources from the literature related to tool integration and those discourses already connected to it, but will return to this discussion at the end of Section 8.

Our approach unfortunately makes the issue of internal validity more complex, since it then amounts to proving that *the literature survey method* is in itself a reasonable means of answering the research questions. In other words, even if all relevant literature is consulted, what if it does not reflect the actual discourse on the essence of tool integration? Therefore, to ensure the validity of the survey, a *data triangulation* (Denzin, 2006) was performed. The categories that had been identified through the survey were used to quantify the answers and questions in a questionnaire issued by the iFEST project in 2010. The focus of the questionnaire was *Best Practices* in embedded system development with regard to tool integration⁵. It was compiled by leading experts in different aspects of tool integration, more precisely researchers and practitioners from the iFEST partners. Most of these researchers and practitioners each have more than 10 years' experience of working with tools in their field. Tool integration was thus considered from in-depth experiences on this subject with regard to requirements engineering, project management, verification and validation techniques, large IT systems, embedded technology, mechatronics, hardware and software co-design, software architectures, formal verification, etc. The design of the questionnaire included both open and closed questions. The questionnaire was sent out to and answered by 23 respondents at 14 European industrial companies. This ensured that answers were provided by (project and product) managers, engineers from different domains, researchers and software architects. The questionnaire is itself therefore an example of a summary of the most important parts of the current industrial discourse. The questionnaire was ensured to be unbiased and cover tool integration relevant to all parts of a product's life-cycle through discussions among the experts aimed at identifying a wide set of activities and stakeholders. Furthermore, the results of the questionnaire were later validated through discussions with other research projects in which the iFEST partners were present.

The results from this comparison are presented at the end of Sections 4, 5, 6 and 7 to support the claim that method bias has not significantly decreased the internal validity.

⁵The observant reader may object to this triangulation on the grounds that iFEST targeted the embedded systems community. The results of the questionnaire may therefore be biased towards that domain, while the intent of this survey is to consider the whole tool integration field. It is then important to remember that the targeted bias is not selection bias but method bias - the requirement on the triangulation is therefore to unearth problems with validity related to across-the-board differences in industrial practice and not necessarily to profile the academic discussion against each and every industrial domain.

3. An Initial Categorization

Brown (1993a), Wicks (2004) and Maalej (2009) have provided valuable categorisations of different strands of tool integration research. However, although they were used as a starting point for this literature survey, ultimately they were too broad for our purpose. More refined (possibly overlapping) sub-categories were instead sought. In addition, seminal papers clearly belonging to a particular strand of research were not necessarily the best for capturing all of these cross-disciplinary strands. A more useful approach was to also survey papers with a focus on technology but partially discussing other strands of research. In short, more “finely grained” categories were required than those provided by Brown (1993a), Wicks (2004) and Maalej (2009). The sources therefore had to be continuously categorized during the reading (and re-reading) to establish a suitable classification scheme. This was done iteratively through discussions among the authors based on excerpts from the sources, with external feedback as noted in the Acknowledgments section. The sources were thus allocated to categories in the process of discovering topics and relationships that were related to the previously mentioned questions linked to the essence of tool integration. The result is a new categorisation scheme that is more fine-grained than already existing ones. The final allocation of sources is presented in Table 1 together with, for convenience, the percentage of sources allocated to each category. The way the topics were handled and the relationships between the categories led to further analysis in four main directions. The reasoning behind this is described below and summarized in Table 2.

Category 1 includes sources that add to the discussion of *non-functional properties* related to tool integration. The typical source touches upon this category by referring to different non-functional properties that are either problematic to or addressed by the discussion in the source. However, usually no in-depth motivation is given for how the non-functional properties were chosen or how they relate to each other. The category is detailed further in Section 4.

Category 2 includes sources that use one or more classification schemes to *structure* (part of) their discussion. Two other categories merit mentioning together with category 2. The first is *category 3*, which includes sources that elaborate on Wasserman’s types of tool integration. The second is *category 6*, which includes those sources that reference a classification scheme other than that of Wasserman (1990). Together these three categories point to the importance given to classification schemes when discussing tool integration, but also to the deep disagreement on exactly what makes a scheme complete. Further research in the direction of these categories is discussed in Section 5.

Category 4 includes sources that present or suggest the implementation of a *framework*, i.e. a realization in software to support tool integration. This is related to two other categories. The first is *category 5*, which includes sources that present some kind of *reference model* related to the design for or evaluation of tool integration. Here the term reference model is used to indicate a more abstract definition, which does not have to be immediately realizable. The second is *category 7*, which includes sources that present some kind of product

related to tool integration (data integration tools, languages for generating tool integration software, etc.). To some extent these categories “blur” into each other, with the allocation to a specific category depending on the emphasis used by the individual authors. With such a high focus on implementation and providing reference models, one would expect the context of tool integration to have been discussed in-depth. Further research in the direction of these categories is discussed in Section 6.

Category 8 and *category 9* are straight-forward. The former includes sources that evaluate standards, reference models or specifications related to tool integration. The latter includes sources that provide an in-depth reporting on empirical data related to tool integration. Both of these categories are rather small, which raised concerns regarding the theoretical underpinning of the targeted discourse. This merited a closer look at which research methods are used in these sources, detailed further in Section 7.

Table 1: Categories

Category 1. Discusses non-functional properties in relation to tool integration (53%)
(Osterweil, 1988), (Clément, 1990), (Earl, 1990), (Brown et al., 1992), (Harrison et al., 1992), (Brown, 1993b), (Brown and Penedo, 1992), (Brown, 1993a), (Long and Morris, 1993), (Belkhatir et al., 1994), (Brown and Penedo, 1994), (Finkelstein et al., 1994), (Sum and Ibold, 1994), (Welsh and Yang, 1994), (Gautier et al., 1995b), (Tilley, 1995), (Tilley and Smith, 1996), (Wasserman, 1996), (Zelkowitz, 1996), (Belkhatir, 1997), (Stavridou, 1999), (Westfechtel, 1999), (Wong, 1999), (Amsden, 2001), (Endig and Jesko, 2001), (Baik et al., 2002), (Becker et al., 2002), (Best et al., 2002), (Losavio et al., 2002), (Schneider and Marquardt, 2002), (Barthelmess, 2003), (Michaud, 2003), (Marquardt and Nagl, 2004), (Tilley et al., 2004), (Dewar, 2005), (Holt et al., 2006), (Kapsammer et al., 2006), (Kramler et al., 2006), (Miatidis, 2007), (Shi, 2007), (Wicks and Dewar, 2007), (Kurniawan and Abramson, 2008), (Hein et al., 2009), (Maalej, 2009), (Biehl et al., 2010b), (Ertürkmen, 2010), (Loutchkina and Nesterov, 2010), (Wende et al., 2010), (Armengaud et al., 2011), (Biehl et al., 2011), (Ceccarelli et al., 2011), (Crelrier et al., 2011), (Kouamou, 2011), (Kurniawan and Abramson, 2011), (Pesola et al., 2011), (Wende et al., 2011), (Armengaud et al., 2012), (Biehl et al., 2012a), (Biehl et al., 2012b), (Jucovschi, 2012), (Skoglund, 2012), (Biehl, 2013)
Category 2. Uses a classification scheme to structure the discussion (44%)
(Clément, 1990), (Brown and McDermid, 1992), (Brown et al., 1992), (Chen and Norman, 1992), (Harrison et al., 1992), (Thomas and Nejme, 1992), (Brielmann et al., 1993), (Brown, 1993a), (Long and Morris, 1993), (Arnold, 1994), (Brown et al., 1994), (Cuthill, 1994), (Bosua and Brinkkemper, 1995), (Gautier et al., 1995a), (Gautier et al., 1995b), (Wasserman, 1995), (Bao and Horowitz, 1996), (Belkhatir, 1997), (Kordon and Mounier, 1997), (Reiss, 1999), (Stavridou, 1999), (Westfechtel, 1999), (Wong, 1999), (Grundy et al., 2000), (Amsden, 2001), (Endig and Jesko, 2001), (Michaud et al., 2001), (Sliski et al., 2001), (Losavio et al., 2002), (Schneider and Marquardt, 2002), (Barthelmess, 2003), (Michaud, 2003), (Stoeckle et al., 2005), (Kienle and Müller, 2007), (Miatidis, 2007), (Shi, 2007), (Bull, 2008), (Izza, 2009), (Biehl, 2010), (Biehl et al., 2010a), (Biehl et al., 2010b), (Ertürkmen, 2010), (Loutchkina and Nesterov, 2010), (Asplund et al., 2011), (Biehl, 2011), (Biehl et al., 2011), (Crelrier et al., 2011), (Nakagawa et al., 2011), (Armengaud et al., 2012), (Biehl et al., 2012a), (Jucovschi, 2012), (Biehl, 2013)
Category 3. Elaborates on Wasserman’s types of tool integration (41%)

(Thomas and Nejme, 1992), (Long and Morris, 1993), (Zelkowitz, 1993), (Brown and Penedo, 1994), (Brown et al., 1994), (Cuthill, 1994), (Nuseibeh, 1994), (Tilley, 1994), (Bosua and Brinkkemper, 1995), (Gautier et al., 1995a), (Gautier et al., 1995b), (Tilley, 1995), (Tilley and Smith, 1996), (Barrett et al., 1996), (Tilley et al., 1996), (Valetto and Kaiser, 1996), (Wasserman, 1996), (Wallace and Wallnau, 1996), (Zelkowitz, 1996), (Pohl and Weidenhaupt, 1997), (Pohl et al., 1999), (Stavridou, 1999), (Wong, 1999), (Tilley, 2000), (Michaud et al., 2001), (Baik et al., 2002), (Becker et al., 2002), (Schneider and Marquardt, 2002), (Stoeckle et al., 2005), (Holt et al., 2006), (Baruzzo and Comini, 2007), (Miatidis, 2007), (Shi, 2007), (Maalej, 2009), (Biehl et al., 2010a), (Biehl et al., 2010b), (Ertürkmen, 2010), (Loutchkina and Nesterov, 2010), (Asplund et al., 2011), (Biehl, 2011), (Biehl et al., 2011), (Crelie et al., 2011), (Jelschen and Winter, 2011), (Koudri et al., 2011), (Biehl et al., 2012a), (Jucovschi, 2012), (Qureshi, 2012), (Biehl, 2013)

Category 4. Presents or suggests a framework implementation (38%)

(Osterweil, 1988), (Ison, 1990), (Chen and Norman, 1992), (Brielmann et al., 1993), (Belkhatir et al., 1994), (Nuseibeh et al., 1994), (Nuseibeh, 1994), (Sum and Ibold, 1994), (Belkhatir and Ahmed-Nacer, 1995), (Gautier et al., 1995a), (Gautier et al., 1995b), (Tilley, 1995), (Emmerich, 1996), (Valetto and Kaiser, 1996), (Belkhatir, 1997), (Kordon and Mounier, 1997), (Pohl and Weidenhaupt, 1997), (Pohl et al., 1999), (Reiss, 1999), (Westfechtel, 1999), (Grundy et al., 2000), (Amsden, 2001), (Endig and Jesko, 2001), (Best et al., 2002), (Marquardt and Nagl, 2004), (Mampilly et al., 2005), (Stoeckle et al., 2005), (Kramler et al., 2006), (Baruzzo and Comini, 2007), (Miatidis, 2007), (Hein et al., 2009), (Wende et al., 2010), (Armengaud et al., 2011), (Biehl, 2011), (Biehl et al., 2011), (Ceccarelli et al., 2011), (Eskeli et al., 2011), (Jelschen and Winter, 2011), (Pesola et al., 2011), (Wende et al., 2011), (Biehl et al., 2012a), (Jucovschi, 2012), (Vasista and AlSudairi, 2012), (Biehl, 2013), (Vasista and AlSudairi, 2013)

Category 5. Presents a reference model (34%)

(Clément, 1990), (Earl, 1990), (Brown and Feiler, 1992), (Brown et al., 1992), (Harrison et al., 1992), (Arnold, 1994), (Brown et al., 1994), (Nuseibeh, 1994), (Welsh and Yang, 1994), (Tilley, 1995), (Barrett et al., 1996), (Chittister and Haimes, 1996), (Tilley et al., 1996), (Zelkowitz, 1996), (Belkhatir, 1997), (Harvey and Marlin, 1997), (Herzog and Törne, 1999), (Losavio et al., 2002), (Schneider and Marquardt, 2002), (Barthelme, 2003), (Michaud, 2003), (Marquardt and Nagl, 2004), (Dewar, 2005), (Kienle and Müller, 2007), (Miatidis, 2007), (Biehl et al., 2010b), (Loutchkina and Nesterov, 2010), (Wende et al., 2010), (Yazbek, 2010), (Armengaud et al., 2011), (Asplund et al., 2011), (Kouamou, 2011), (Koudri et al., 2011), (Kurniawan and Abramson, 2011), (Nakagawa et al., 2011), (Wende et al., 2011), (Jucovschi, 2012), (Vasista and AlSudairi, 2012), (Biehl, 2013), (Vasista and AlSudairi, 2013)

Category 6. Uses a classification scheme other than Wasserman's (28%)

(Clément, 1990), (Earl, 1990), (Ison, 1990), (Brown and McDermid, 1992), (Brown et al., 1992), (Chen and Norman, 1992), (Harrison et al., 1992), (Brielmann et al., 1993), (Brown, 1993a), (Arnold, 1994), (Cuthill, 1994), (Sum and Ibold, 1994), (Gautier et al., 1995b), (Chittister and Haimes, 1996), (Zelkowitz, 1996), (Reiss, 1999), (Stavridou, 1999), (Westfechtel, 1999), (Amsden, 2001), (Endig and Jesko, 2001), (Sliski et al., 2001), (Losavio et al., 2002), (Barthelme, 2003), (Michaud, 2003), (Mampilly et al., 2005), (Kienle and Müller, 2007), (Miatidis, 2007), (Bull, 2008), (Izza, 2009), (Loutchkina and Nesterov, 2010), (Wende et al., 2010), (Yazbek, 2010), (Wende et al., 2011)

Category 7. Presents a product related to tool integration (14%)

(Arnold, 1994), (Tilley, 1994), (Daberitz and Kelter, 1995), (Wong, 1999), (Michaud et al., 2001), (Becker et al., 2002), (Grundy et al., 2006), (Bull, 2008), (de Alwis and Murphy, 2008), (Kurniawan and Abramson, 2008), (Ertürkmen, 2010), (Crelieu et al., 2011), (Kurniawan and Abramson, 2011), (Armengaud et al., 2012), (Biehl et al., 2012b), (Qureshi, 2012)
Category 8. Evaluates standards, reference models or specifications related to tool integration (8%)
(Brown and Feiler, 1992), (Brown, 1993a), (Long and Morris, 1993), (Zelkowitz, 1993), (Cuthill, 1994), (Barrett et al., 1996), (Wallace and Wallnau, 1996), (Shi, 2007), (Skoglund, 2012)
Category 9. In-depth reporting on empirical data related to tool integration (1%)
(Maalej, 2009)
Not included
(Arnold, 1995), (Woodside, 1995), (Gimenes et al., 1997), (Conradi and Westfechtel, 1998), (Bruegge et al., 2008), (Kurniawan and Abramson, 2008), (Carbone et al., 2010), (Haase and Nagl, 2011), (Zhang et al., 2011), (Kääriäinen, 2011), (Biehl et al., 2012c), (Gu, 2012)

Table 2: Further Analysis based on Topics and Relationships

Category	Motivation	Discussed Further in Section
1	Non-functional properties often referred to, but in vague terms.	4
2, 3 and 6	Large and varied amount of classification schemes.	5
4, 5 and 7	High focus on frameworks, reference models and associated products (concepts that to some extent “blur”), implying a well known system context.	6
8 and 9	Small effort on evaluation and empirical data, raising concerns with regard to the theory underlying the discussion.	7

4. Tool Integration and Non-functional Properties

The type of non-functional properties and the way in which they have been discussed is detailed further in this section. To support the internal validity of these findings, the most frequently mentioned non-functional properties are then compared to those frequently occurring in the iFEST questionnaire.

Figure 1: Occurrences of Nonfunctional Properties, Percentages

Figure 2: Occurrences of the Top Six Properties over Time

4.1. Findings

As seen in Table 1, a majority of the sources touch upon the non-functional properties of tool integration. These properties are by nature not easily defined, with most sources not offering any clear definitions, priority order or discussion of how these properties relate. An examination of the definitions of the properties given by the sources also shows that different names are often used to refer to the same property, and vice versa - that the same name can actually refer to different properties. Nevertheless, even after merging the cases when the sources simply seem to be using different terminologies, one is still left with an impressive 30 *different* non-functional properties. Of these properties the 10 most often mentioned make up 73% of the occurrences counted, with the top six properties on their own claiming similar shares out of 58% of the occurrences counted (see Figure 1). Furthermore, when the top six properties are laid out in a graph showing their occurrences across the last two decades, each show an even distribution over time (see Figure 2).

In other words, throughout the discourse a few non-functional properties have stood out as more important to discuss than others. The top six of these non-functional properties are discussed in more detail below.

- *Flexibility* is the ease of which a support environment can be adapted. A

requirement for flexibility can stem from the need to handle early misunderstandings of the requirements on a support environment (Osterweil, 1988), seamlessly introduce novel features (Tilley and Smith, 1996) and support cost savings (Welsh and Yang, 1994). Flexibility is therefore discussed both in regard to the parts that make up a support environment and a support environment seen as a whole. The former discussion has for instance touched upon the interchangeability of tools (Jucovschi, 2012), while the latter has dealt with how to adapt to the specific conditions of different companies (Endig and Jesko, 2001) or even different application domains (Becker et al., 2002).

- *Scalability* is the degree to which a support environment can be treated, and be expected to behave, similarly, regardless of changes to the number of parts that it consists of. Lack of scalability support can show itself through inconsistencies (Finkelstein et al., 1994) or by end users being overwhelmed by too much information (Tilley, 1995). Scalability can be discussed in relation to parts of support environments, such as standard exchange formats (Holt et al., 2006). However, some regard this property as primarily related to the whole (Miatidis, 2007). Solutions for achieving scalability have been put forward in both of these discussions. Scalability has for instance been claimed to be a benefit of component-based architectures (Michaud, 2003). However, no strong case has been made for a relationship between these solutions and certain high level issues linked to scalability, such as the flaws in cognitive support and knowledge transfer mentioned by Brown and Penedo (1994).
- *Cost* is the impact that support environments have on financial matters. When discussed in a narrow sense, this relates to the direct costs of designing, constructing and deploying support environments (Brown, 1993b), but also how to measure their impact (Baik et al., 2002). When discussed in a broader sense, cost relates to the implications of the business context of related stakeholders. For instance, different business models of tool vendors have been discussed both as a limitation to which tools are integrated (Brown et al., 1992) and as a prerequisite for allowing any major benefits to be gained by deploying support environments (Tilley et al., 2004). In this discussion, the use of support environments has been mentioned as commonly based on business goals (Wicks and Dewar, 2007) or even as a critical part of achieving business success (Marquardt and Nagl, 2004).
- *Evolvability* is the potential of a support environment to facilitate change over time. This is discussed from two perspectives. Firstly, in regard to changes inside the support environment itself. For instance, how the availability of evolvable support environments can be important in ensuring the success of new technology (Dewar, 2005) or how evolution needs to be possible to drive and direct (Osterweil, 1988). Secondly, in regard to how a support environment can be used to facilitate product evolution when used during development (Belkhatir et al., 1994; Belkhatir, 1997).

Additionally, evolvability is also discussed in relation to whole support environments and the parts they are made up of: the former in regard to how economical and political influences can impact the possibility of support environments to evolve (Earl, 1990); and the latter in regard to which technology is best suited for facilitating change (Michaud, 2003).

- *Efficiency* is discussed in conjunction with tool integration with regards to both the technological performance of support environments (Shi, 2007) and how organizations utilizing support environments can perform better (Armengaud et al., 2012). The latter, broader focus is naturally linked to the discussion on automation of tedious, repetitive tasks (Westfechtel, 1999).
- The *degree of standardization* is the degree to which a support environment conforms to readily available specifications (even though the distribution of the specifications might be limited to, for instance, those that have paid a fee). The impact of standards may be on details and on broader concerns, since different standards target different levels of abstraction. The motivations given for an increased degree of standardization are therefore diverse, such as enabling adaption to different domains (Sum and Ibold, 1994), being a prerequisite to dealing with tool integration product life-cycle issues (Schneider and Marquardt, 2002), avoiding tool vendor lock-in (Ertürkmen, 2010), and so on.

Even if the provided definitions help differ between the mentioned non-functional properties, it does not automatically mean that it is valuable to do so. In the case of support environments these non-functional properties can be too interconnected to allow for them to be studied in isolation. In Subsection 8.1 we provide a number of high level research questions connected to these non-functional properties for future research. An added benefit of pursuing several of these questions in parallel would be the possibility to understand whether there are common obstacles, such as interconnectivity, to answering this type of questions.

4.2. Internal Validity of the Findings

Of the non-functional properties mentioned in relation to tool integration in the iFEST questionnaire, the top 5 are the degree of standardization (31%), efficiency (22%), flexibility (12%), support for evolution (9%) and cost (8%). These are also the only ones with a considerable share of the total. Scalability is not mentioned at all. It is reasonable to assume that method bias has not lead to incomplete results, but that scalability is perhaps emphasized more in academic discourse than in industrial settings.

5. Integration Types

Based on the importance given to classification schemes by the sources, this section starts by describing the way the sources have used and elaborated on

Figure 3: Use of the Types of Tool Integration, Percentages of Relevant Sources

Wasserman’s classification scheme (Wasserman, 1990), and then contrasts it with other approaches that describe tool integration at a high level of abstraction. To support the internal validity of these findings, the classification scheme introduced by Wasserman (1990) is also related to the iFEST questionnaire.

5.1. Findings

Wasserman (1990) introduced 5 types of tool integration, namely *Control*, *Data*, *Platform*, *Presentation* and *Process Integration*. These are described as “*dimensions*”, i.e. separate, unconnected concerns. The separation between the types of tool integration rests on relating them to *different* kinds of supporting tools and mechanisms. These types of tool integration are by far the most popular idea brought forward by Wasserman (1990). While 44% actively use Wasserman’s types of tool integration to structure their discussion, a further 22% mention them. Out of the 28% that make use of other classification schemes, 40% do so in combination with Wasserman’s types of tool integration.

However, a deeper investigation of the 66% that make use of Wasserman’s classification scheme, summarized in Figure 3, shows that the use is not coherent. Many of these sources elaborate on the meaning of the different types or choose freely which of them to take into account.

In the five subsequent subsections we start by describing each of these types as they were presented by Wasserman (1990) and then go through how other sources have viewed them differently. Thereafter we discuss other approaches to defining types of tool integration and close by discussing the internal validity of the results.

5.2. Control Integration

Wasserman (1990) described control integration as the ability of tools to notify each other of events and activate each other under program control.

This definition prevails throughout the discourse, although a significant number of the surveyed sources try to add on different process aspects such as process management (Zelkowitz, 1993), coordination and synchronization to enable co-operation (Nuseibeh, 1994) and the ability of tools to notify *users* (Biehl, 2011). Several sources also try to clarify control integration by tying its definition to

tool *functionality*, for instance by referring to the use and provision of functionality (Thomas and Nejme, 1992), or the intent to combine functionality (Brown and Penedo, 1994).

5.3. Data Integration

Wasserman (1990) described data integration as the ability of tools to share data with each other and manage the relationships among data objects produced by each other.

This definition mostly prevails throughout the discourse. A few sources limit the definition, for instance by only referring to data sharing (Brown et al., 1994). Most sources try to clarify the definition through examples, e.g. data persistence (Stavridou, 1999), syntax (Biehl et al., 2010b), semantics (Schneider and Marquardt, 2002), consistency (Thomas and Nejme, 1992) and traceability (Koudri et al., 2011).

5.4. Platform Integration

Wasserman (1990) described platform integration as the set of system services that provide network and operating systems transparency to tools and tool frameworks.

This definition almost entirely prevails throughout the discourse, mostly in vague references to “common platforms”. The only elaboration makes the definition wider by including all common parts of an environment, rather than just services (Asplund et al., 2011).

5.5. Presentation Integration

Wasserman (1990) described presentation integration as the set of services and guidelines that allow tools to achieve a common “*look and feel*” from the user’s perspective.

This definition prevails throughout the discourse, but with several interesting points raised by those that do not adhere to it. Some sources add to the definition by including user interface sophistication (Brown et al., 1994; Stavridou, 1999) or interaction paradigms (Thomas and Nejme, 1992). Others entirely reject the view that the goal of presentation integration is always to produce a *uniform* user interface (Wong, 1999; Tilley, 2000; Asplund et al., 2011). These instead argue that the focus should be changed to the integration of the *different users* to the tools via their user interfaces. This means that a tool should be able to present different user interfaces to users with different professional roles, backgrounds, knowledge, purposes or visualization preferences. The goal of presentation integration would then be to facilitate the correct matching of user interfaces to the different users. The idea that presentation integration is by necessity a separate concern is even challenged by Stoeckle et al. (2005), since visualization notations may, through control or data integration, achieve the same result as a common “look and feel”.

5.6. Process Integration

Wasserman (1990) described process integration as the integration of process management tools with other tools in the support environment.

In a later paper Wasserman himself redefined this type of tool integration using the much wider “*linkage between tool usage and the software development process*” (Wasserman, 1996). In fact, many of the sources in the discourse have challenged Wasserman’s original definition. And in some cases, the later definition is used even though the sources refer to the original paper (Thomas and Nejme, 1992; Baruzzo and Comini, 2007).

The most common way of defining process integration is to simply state that it concerns ensuring a proper match between processes and tool integration technology (Long and Morris, 1993). Other common definitions tie process integration to the definition and integration of process models (Nuseibeh, 1994), the awareness and enforcing of process constraints on tools based on user roles and process states (Kordon and Mounier, 1997), or both (Gautier et al., 1995a). Additionally, while the perspective of most sources is that processes should determine which tools are used and how they behave, a few sources acknowledge the fact that legacy tools and tool integration technology might enforce a particular structure on workflows (Marquardt and Nagl, 2004; Biehl et al., 2010a).

5.7. Other Approaches

A classification scheme is a collection of conceptual categories that highlight important, distinct parts of a subject under discussion. As such it should be detailed enough to allow reasoning about (a part of) a subject, but also abstract enough not to unnecessarily confuse the discussion. Classification schemes are therefore closely related to reference models, the difference being that reference models even though they do not have to be immediately realizable should at least be detailed enough to allow independent work in relation to the subject (for instance on standards (Earl, 1990)). Not surprisingly one other approach to defining types of tool integration is therefore to link Wasserman’s types of tool integration to different parts of a reference model, for example as done by Brown et al. (1994) in relation to a reference model which divides support environments into three levels, namely the *mechanism* (how are the components in the support environment connected), *service* (what does the support environment provide) and *process levels* (which activities does the support environment support). Control, data, platform and presentation integration are allocated to the mechanism level, while process integration is allocated to the process level. The same source also raises doubts as to whether the different types of integration are in fact “dimensions”, proposing that as mechanisms become increasingly sophisticated, the boundaries between the different types may blur. Later Losavio further detailed this reference model by differentiating between integration targeting the internal organization, external entities and management (Losavio et al., 2002).

The most common other type of approach, however, is a classification scheme that differs from Wasserman’s only in the use of more categories within a type

of tool integration or by drawing the boundaries between them differently. An example of the former is Brown and McDermid (1992) equating tool integration to data integration, which is then further divided into the *carrier*, *lexical*, *syntactic*, *semantic* and *method levels*. An example of the latter is proposed by Brielmann et al. (1993) in the form of the three separate dimensions of *control*, *data* and *user interface integration*. These three dimensions are a combination of Wasserman’s control and process integration, a combination of Wasserman’s data and (part of) platform integration, and the same as Wasserman’s presentation integration, respectively. Data integration is the most commonly discussed type of integration in these approaches, followed by presentation, control and process (in that order).

Other approaches add to Wasserman’s classification scheme. Zelkowitz, for example, mentions three *notions* of tool integration that affect the design of support environments (Zelkowitz, 1996), namely the *conceptual* (a shared philosophy in regard to the interaction of support environment components), the *architectural* (how support environment components are constructed to interact) and the *physical integration* (the interaction of the actual instances of support environment components).

A few other approaches make use of wholly different types. The most obvious example in the sources is Wende et al. (2010) using a classification scheme that measures support environments according to two dimensions: *extensibility*, which in this source is a measure of *customizability*; and *guidance*, which is a measure of how well the process of customization of the support environment is supported. This scheme was later further extended by adding *reuse*, which is a measure of the reuse of shared platform functions between different support environments (Wende et al., 2011).

5.8. Internal Validity of the Findings

All references in the iFEST questionnaire that can be related to abstract categories of tool integration fall within the original types of tool integration suggested by Wasserman. Interestingly enough, of these references most relate to data (54%) and platform (18%) integration. Again, it is reasonable to assume that method bias has not lead to incomplete results. The high focus on *data* and *platform* integration in the questionnaire, coupled with the survey sources mostly using them “as is” or ignoring them, is an indication that these are the least controversial categories in the discourse.

6. The System Context

Based on the high focus on implementation and providing reference models, this section takes a closer look at the system context used by the sources when discussing tool integration. This results in an in-depth discussion of stakeholders relevant to tool integration, since most sources show no conclusive evidence of a well understood system context. To support the internal validity of these findings the stakeholders mentioned by the sources are also related to the iFEST questionnaire.

Figure 4: Overall Contexts, Percentages

6.1. Findings

Figure 4 shows the distribution of the overall contexts that the sources use to frame their discussion. A few sources *generalize*, meaning that they imply that tool integration can be treated in a similar way across all contexts. More sources discuss tool integration as framed by *systems development*, a context that can be further divided into two groups. The first includes those sources that have a *narrow* focus on only the development phase or provide no detailed definition of systems development. The second includes those sources that at least mention a *broad* view of other life-cycle phases of systems development, such as maintenance, production, decommission, etc. Similar groups of sources can be found that further restrict themselves to *software development*. Finally, one group consists of sources that use a specific *application domain* as a context for discussing tool integration, with the most often used domains being software re-engineering (22%), enterprise applications (19%) and chemical engineering (15%).

It is clear that development is the most common context envisioned when discussing the essence of tool integration, either in a narrow (39%) or a broad (32%) sense. This is not a very strong finding, considering that the basis for the survey is a paper in the software engineering field (Wasserman, 1990). However, the initially general or non-existent description of the system context in most sources suggests that it is an unimportant, obvious or overlooked factor. That the system context is unimportant is belied by the other findings in this report, such as the higher level at which some non-functional properties are discussed (see Section 4) and the many sources which elaborate on process integration (see Section 5). To ascertain whether the details of this factor are obvious to or overlooked by those involved in the discourse we further studied the relevant sources. A closer look revealed that only 13% (of all sources) provide a more in-depth discussion of the context of tool integration. Of these, the majority either discuss different scales of organization (20%) or different stakeholders (60%). The former relates to discussing tool integration as important to

Figure 5: Stakeholders, Percentages

the individual, the team, the organization and interorganizational relationships. The latter is summarized in Figure 5 by listing the top 5 stakeholders being discussed. These are the *tool developers*, *management* (in any related organization), *support environment customizers* (those putting together a particular instance or providing support for it), *support environment designers* (those developing the basic support environment software) and *end users* (in the role of users and customers).

This one-sided focus on a narrow set of stakeholders, important to the implementation of a support environment, points at the system context being overlooked rather than obvious to those involved in the discourse. An in-depth discussion of the continuous evolution of a support environment throughout its lifetime would require one to consider the much larger set of stakeholders involved in relevant decision-making, standardization attempts and maintenance.

6.2. Internal Validity of the Findings

End users (in the role of users (76%) and customers (16%)) are the only stakeholders mentioned in the iFEST questionnaire to any great degree. It is reasonable to assume that method bias has not lead to an incorrect perception of a high focus on end users, but that the focus on support environment designers is probably higher in the academic discourse than in industrial settings.

7. Research Methods

Based on there being few sources that provide in-depth reporting on empirical evidence related to tool integration, this section studies the research methods employed by the different sources. The problem in establishing the validity of the findings for any part of the discourse that takes place outside the academical literature is then highlighted in the second subsection.

7.1. Findings

As seen in Table 1, there is a lack of published in-depth empirical data in relation to the discussion on the essence of tool integration. 8 different research

Figure 6: Research Methods, Percentages

methods could be identified when the sources were studied further. The summary of how many sources employed each research method can be found in Figure 6.

Expert knowledge is a part of all scientific inquiry, but the largest group identified consists of those sources that *only* make use of expert knowledge. This is not necessarily a problem. A reference model or a summary of the State of the Art does not have to include a solid empirical base to be useful, but rather rests on the reasoning of those experts involved in writing it (the paper by Brown (1993b) is a good example of this). However, several of the sources included in this group simply did not provide a description of the methods used, which has to be seen as problematic with regards to judging the validity of their findings. The second largest group consists of those sources that make use of *case studies*, i.e. studies of a limited number of instances of a phenomena. Even though this research method has been criticized over how to ensure that the results are generalizable, this is also not necessarily problematic. Only one of these case studies however, was a critical case study, i.e. set up specifically so that it could answer a particular research question. All of the others were open-ended, exploratory case studies. All of the other groups are formed by research methods that were only used by 1-3% of the sources. These include *interviews*, *questionnaires*, *mathematical proofs*, *literature surveys*, *content analyses* and *experiments*.

Most of these groupings are self-explanatory, but it should be noted that only sources providing a description of a *literature survey method* were included in the literature survey group. In other words, even though most academic papers include some part that discusses relevant literature, this grouping only included those sources that had the explicit purpose to survey a large amount of literature based on well defined selection criteria.

7.2. Internal Validity of the Findings

Research methods are not discussed in the iFEST questionnaire at all. This part of the paper therefore only provides input to the conclusions in regard to

the historical, academic discourse on tool integration. The true distribution of methods employed to gain knowledge on the essence of tool integration may in fact be different, for instance if industrial companies perform a lot of experiments that are never reported.

8. Discussion

This section discusses the discourse on the essence of tool integration, its context and the implications in light of the findings presented in the previous sections. New research questions are put forward based on the non-functional properties mentioned throughout the discourse, something which also has implications on the study of stakeholders relevant for tool integration. The problem of evolving Wasserman’s types of tool integration is discussed, with two frequently mentioned redefinitions highlighted. Potentially problematic effects due to the set of research methods currently employed in the discourse are highlighted, and a proposal on how to change this set outlined. The section then closes by discussing the need to further investigate disparate discourses and highlighting the importance of open communities in relation to this.

The observant reader may notice that the order of the subsections below do not directly match that implied by the order of the previous sections. While the order of the previous sections follow the size of the initial categories discussed within them, the order of the subsections below was chosen to ease the flow of the discussion.

8.1. *A Discourse Focused on Details*

The non-functional properties identified in Section 4 to some extent provide a previously missing prioritization order. This order is not an objective measure, but rather reflects what has been seen as important by many taking part in the discourse on the essence of tool integration.

More importantly, while the overall discourse on tool integration largely focuses on mechanisms (Wicks and Dewar, 2007), the studied discussion related to non-functional properties indicates that the focus should be less on details and more on issues at higher levels. Based on the identified properties, for instance, the following (research) questions merit attention in future research:

- Do different application domains put different requirements on tool integration?
- How can tool integration support stakeholders in interacting not only with all data in a support environment scaled to modern situations, but with the right data?
- How can business models help drive the deployment of tool integration, not just lock users to a single vendor?

- What is the impact of a technological shift, such as the recent decline in the use of Java in favour of RESTful web services, on an organization employing a modern support environment?
- When does automating a series of tasks actually provide efficiency gains?
- How is tool integration standardized? In which standards and by which standardization bodies?

These are just representative examples for each of the non-functional properties found in Section 4, more (research) questions akin to these need to be formulated and related research areas inventoried for relevant knowledge.

8.2. *An Increased Set of Stakeholders*

Research questions with a wider scope will require knowledge of more stakeholders than those found in the discourse on the essence of tool integration so far. As shown in Section 6, currently end users receive most of the attention in the discourse even if support environment designers also receive a fair share of the focus in the literature. Based on the questions mentioned in the previous subsection one can, for instance, identify the following stakeholders as being of interest to study further:

- *Application domain experts.* As an example, software engineers commonly receive abstract product requirements and implement the design directly in code. Hardware engineers by contrast spend a lot of time in various design tools representing different levels of abstraction. These different domains therefore put different demands on when and why tool integration should be deployed to support development.
- *Project managers.* While project managers are end users, they are presumably not only interested in moving data from one tool to another. The possibility to link to data sources throughout the product life-cycle to enable analysis is clear, but to enable any benefits the set of analyses of interest needs to be known.
- *Managers.* Managers ultimately decide which solutions are put in place, but the metrics needed to support this decision-making have so far been elusive. Currently the parameters that factor into these equations are largely unknown.
- *Support environment administrators.* The deployment of several different systems, or difficulties in configuring systems, impacts administrators to a large degree. This leads to resistance in deploying large-scale support environments and favours ad hoc tool integration. For large-scale support environments to even be considered, the prioritizations of those that will maintain them also needs to be considered.

- *Customers.* A modern support environment may enable an increased pace of product releases, something which is already a reality in some software engineering domains. While changes to release frequency may be beneficial in many ways, customers may have difficulties in adopting and become wary if they are not appropriately consulted.
- *Standardization organizations.* Even large research projects frequently have problems with standardizing their findings. Standardization organizations are entities of their own and ensuring their approval and use of research findings is not straight-forward.

The choice of which stakeholders to study in the end also depends on the context that is being studied. However, currently most researchers feel no need to detail the overall context of their research when discussing the essence of tool integration. As described in Section 6, they either contend that tool integration is straight-forward to generalize or avoid the question altogether by describing their engineering context in vague terms. However, if these “generic” solutions are easily applicable in all contexts, how come the main part of most industrial support environments are instead made up of ad hoc solutions? In reality most of the specialized requirements of different application domains remain unknown, meaning that all claims of generic solutions cannot be said to be based on empirical evidence. In the end the primary motivation for detailing the relevant stakeholders might be to expose the way their motivations differ from case to case, thereby paving the way for configurable tool integration that can enable truly generic support environments.

8.3. A Note about the Types of Tool Integration

The types of tool integration originally identified by Wasserman are also likely to be treated similarly in relation to a wider context and a larger set of stakeholders. At a higher level of abstraction they combine and elaborate freely into a flexible checklist for discussion of the essence of tool integration. However, although this flexibility has meant that the types have been used extensively, it also means that constant effort is required to maintain a place in the discourse for proposed changes to their definitions. For instance, process integration, as shown in Figure 3, has required constant redefinitions throughout the discourse. This regardless of the fact that even Wasserman later chose to use a wider definition of this type (Wasserman, 1996).

Two proposed redefinitions are frequently mentioned but have failed to be used by those not explicitly interested in these specific parts of the discussion:

- the notion that the “dimensions” are not independent. With such wide definitions in use, these types are commonly discussed as if they were *de facto* related. This has implications primarily on presentation and process integration, since these are dependent on the other categories to a higher degree (Asplund et al., 2011). A consequence of this view is that not only the types should be discussed in regard to proposed solutions, but also the different dependencies that exist between them.

- the view that presentation integration should not target uniform environments, but rather enable customized Graphical User Interfaces (GUIs) for users with different needs.

Hopefully this paper can in some way facilitate a wider use of these redefinitions.

8.4. Research Methods

The most commonly used research methods in the discourse on the essence of tool integration are expert knowledge and exploratory case studies. As discussed in Section 7, this is not necessarily a problem. However, when the research field is permeated by these two research methods and by studies that do not disclose their research designs to any high degree, *then* it is natural to ask oneself how sure one can be in the validity of the accepted “truths” of the field. *Generalizability* in particular is not ensured by the repetition of the same opinion by many experts, or by the success of a particular solution in several cases.

If more research methods are required to strengthen the claim of validity, which should then be chosen? It could be argued that just using *more* research methods to encourage the development of different views in a research field is valuable in itself, but if care is not taken then there are potential risks, for instance incoherent theory development (Blessing and Chakrabarti, 2009). Some research methods are also potentially very fruitful, but difficult to use within software engineering (Fenton et al., 1994). A more plausible way forward is to design a change to the set of research methods based on the distinguishing characteristics of the research field itself.

One such defining characteristic is the strong focus on software design solutions by researchers in the field. This focus is shown by previous studies (Wicks and Dewar, 2007), the focus on implementations (see Section 3) and the choice of stakeholders in the academic discourse (see Section 6). One conclusion is that research into tool integration is performed just as much with the intent to invent and present new software designs as to provide knowledge on the phenomena of tool integration. A start towards a changed set of research methods is most likely to be successful if it can be motivated in combination with such an intention.

- A first step by researchers into tool integration should be to consider a shift from exploratory case studies to critical case studies. In other words, more cases should be chosen with care for how easily generalizable their findings are. Flyvbjerg (2011) gives the example of how measuring the impact of handling organic solvents on workers at an enterprise that *rigorously* follows safety regulations might be generalizable to workers at enterprises that are not as rigorous. Similarly, it may be easier to claim valid findings if a new technology for evolvable and scalable support environments is deployed at an organization undergoing significant growth, rather than at an organization that remains static for the duration of the study.

- Considering the large efforts put into implementation, the design of support environments can both attract researchers from other research fields and achieve synergy effects together with them. If a new mechanism for support environments is constructed, then its impact on software customization, data mining, systems engineering and different application domains could be studied in parallel. This would provide the possibility to set up *theory triangulation* studies (Denzin, 2006), which could foster the use of several research methods in *parallel mixed method studies* (Tashakkori and Teddlie, 1998). A reasonable assumption is that interviews and content analysis can play a role in such studies, at least in the many cases when the system context is given by such abstract and complex activities as system and software engineering (see Section 6). For example, interviews can be used to collect data on the efficiency of different visualization techniques, with the execution times of the underlying technology simultaneously recorded; defect data can be gathered and statistically analysed, with engineers interviewed on the likely origin of quality problems.

Besides helping to ensure validity, diversifying the employed research methods in this way would be beneficial from the perspective of tool integration researchers in other ways. With regard to the critical case studies we argue that it would lead to more focused field studies. With regard to theory triangulation we argue that it would lead to a quicker uptake of State of the Art findings from other research fields. Especially better focused field studies would make it possible to answer such high level research questions as those found in Subsection 8.1. Furthermore, both of these benefits are important not just to academia, but also to the industry. Testing unproven technology quickly and with the intent to identify the boundaries within which it is realistic to use should lead to knowledge that is more valuable to decision makers.

8.5. *Disparate Discourses*

When scanning through the final set of sources, we can identify several expected connections, e.g. the Model-Driven Engineering (MDE) discourse (e.g. through the paper by Kapsammer et al. (2006)). However, there are also some that we did not expect, which connect the targeted discourse to, for example, chemical engineering (e.g. through Marquardt and Nagl (2004)), and software re-engineering (e.g. through Tilley (1995)). As mentioned in the discussion on validity in Subsection 2.2, there may also be relevant findings in unconnected discourses that are difficult to tie to the targeted discourse due to differences in terminologies, research methods and models. We can identify at least one that has a high potential of containing findings relevant for tool integration - namely the discourse on ISO 10303 (informally known as the *STandard for the Exchange of Product model data (STEP) standard*).

The discourse on STEP is primarily focused on data integration (Pratt, 2005), as is the previously mentioned MDE discourse (Giese et al., 2010). This

paper shows that the tool integration discourse is not limited to low level implementation details, and the STEP discourse has similar implications. High level issues discussed in the STEP discourse include, for example, business drivers, high level stakeholders and different application domains (Fowler, 1995). It is particularly interesting that this disparate discourse focuses on a standard. We also work with safety standards, such as IEC 61508 (2010), ISO 26262 (2011) and DO-178C (2011a). Changes in industrial practices have required these standards to take tools and tool integration more and more into account. For instance, DO-330 (2011b), which was one of the first released supplements of DO-178C, focuses solely on software tool qualification. Although there is a connection between the targeted discourse and the discourse on these standards (e.g. through (Armengaud et al., 2012)), it is weak.

Further research is required to investigate the implications in these and other disparate or weakly connected discourses. Seeing that standardization is an important topic in the targeted discourse, communication of experiences and knowledge across open communities employed in the context of (pre-)standardization could be especially fruitful. Not only would this communication create opportunities for theory triangulation (see the previous subsection), but it would also offer opportunities to observe and interact with a larger set of stakeholders in their particular system contexts.

9. Conclusions

By surveying sources in the strands of research focusing on the essence of tool integration, six non-functional properties that are treated as especially important have been identified: flexibility, scalability, cost, evolvability, efficiency and the degree of standardization. In the surveyed sources these properties are discussed both with a narrow focus on technology and a broader focus on how a support environment as a whole impacts its environment. However, this environment, or system context, is vaguely defined, with important stakeholders only discussed in passing if at all.

The discussion in the targeted discourse related to the six non-functional properties points at the lack of research into how support environments impact and are impacted by high level issues, such as differences between application domains, business models and organizational efficiency. Redeeming this lack will require a substantial effort to identify the motivations of a larger set of stakeholders and how these motivations differ between different system contexts.

Wasserman’s types of tool integration are likely to continue to be used as a flexible checklist, regardless of whether this lack is redeemed or not. However, this checklist would benefit from two changes. Firstly, the notion of an independence between the types needs to be rejected, and each of the dependencies that exists between the types incorporated into the checklist. Secondly, presentation integration should be redefined, focusing on how different users interact with GUIs rather than how GUIs can be made to look the same.

Furthermore, a summary of the research methods used throughout the targeted discourse shows a large proportion of studies that use exploratory case

studies or only expert knowledge. This, together with the high number of studies that do not provide details on their research design, motivates a changed set of research methods to ensure validity. A shift towards critical case studies and theory triangulation stands a good chance of acceptance and fruitfulness, based on the common intention to invent and present new software design in the discourse.

Finally, there are discourses that are separate from or weakly connected to the targeted discourse, but which have a high probability of containing relevant findings. The examples highlighted by us are standardization discourses, implying that open communities employed in the context of (pre-)standardization could be especially important in furthering the discourse on the essence of tool integration.

Acknowledgments

Special thanks go to Vicki Derbyshire for her help in proofreading this paper and Jad El-khoury for his advice on how to improve certain parts of it.

References

- de Alwis, B., Murphy, G.C., 2008. Answering conceptual queries with Ferret, in: ACM/IEEE 30th International Conference on Software Engineering, 2008. ICSE '08., pp. 21–30. doi:10.1145/1368088.1368092.
- Amsden, J., 2001. Levels of integration: Five ways you can integrate with the eclipse platform. Eclipse Corner.
- Armengaud, E., Biehl, M., Bourrouilh, Q., Breunig, M., Farfeleder, S., Hein, C., Oertel, M., Wallner, A., Zoier, M., 2012. Integrated tool-chain for improving traceability during the development of automotive systems, in: Proceedings of the 2012 Embedded Real Time Software and Systems Conference.
- Armengaud, E., Zoier, M., Baumgart, A., Biehl, M., Chen, D., Griessnig, G., Hein, C., Ritter, T., Kolagari, R.T., 2011. Model-based toolchain for the efficient development of safety-relevant automotive embedded systems, in: Proceedings of the SAE 2011 World Congress & Exhibition. URL: <http://papers.sae.org/2011-01-0056>.
- Arnold, J.E., 1994. Toward collaborative software processes, in: Proceedings of the Ninth International Software Process Workshop, 1994., pp. 107–109. doi:10.1109/ISPW.1994.512777.
- Arnold, J.E., 1995. Control integration: A briefly annotated bibliography. ACM SIGSOFT Software Engineering Notes 20, 62–67. URL: <http://doi.acm.org/10.1145/224155.224162>, doi:10.1145/224155.224162.

- Asplund, F., Biehl, M., El-khoury, J., Törnngren, M., 2011. Tool integration beyond Wasserman, in: *Advanced Information Systems Engineering Workshops: CAiSE 2011 International Workshops*, London, UK, June 20-24, 2011, Proceedings, pp. 270–281.
- Association for Computing Machinery, 2013. ACM digital library. URL: <http://dl.acm.org/>.
- Baik, J., Boehm, B., Steece, B.M., 2002. Disaggregating and calibrating the CASE tool variable in COCOMO ii. *IEEE Transactions on Software Engineering* 28, 1009–1022.
- Bao, Y., Horowitz, E., 1996. A new approach to software tool interoperability, in: *Proceedings of the 1996 ACM symposium on Applied Computing*, ACM, New York, NY, USA. pp. 500–509. URL: <http://doi.acm.org/10.1145/331119.331432>, doi:10.1145/331119.331432.
- Barrett, D.J., Clarke, L.A., Tarr, P.L., Wise, A.E., 1996. A framework for event-based software integration. *ACM Transactions on Software Engineering and Methodology* 5, 378–421. URL: <http://doi.acm.org/10.1145/235321.235324>, doi:10.1145/235321.235324.
- Barthelmeß, P., 2003. Collaboration and coordination in process-centered software development environments: A review of the literature. *Information and Software Technology* 45, 911–928. URL: <http://www.sciencedirect.com/science/article/pii/S0950584903000910>, doi:[http://dx.doi.org/10.1016/S0950-5849\(03\)00091-0](http://dx.doi.org/10.1016/S0950-5849(03)00091-0).
- Baruzzo, A., Comini, M., 2007. Toward a unified framework for quality and consistency verification of UML models. URL: <http://www.dimi.uniud.it/comini/Papers/FrameworkQualConsUML/FrameworkQualConsUML.pdf>.
- Becker, S., Haase, T., Westfechtel, B., Wilhelms, J., 2002. Integration tools supporting cooperative development processes in chemical engineering, in: *Proceedings of the 6th Conference on Integrated Design & Process Technology*.
- Belkhatir, N., 1997. An object oriented framework for distributed, interoperable process engineering environments, in: *Technology of Object-Oriented Languages and Systems*, 1997. TOOLS 23. Proceedings, pp. 318–328. doi:10.1109/TOOLS.1997.654741.
- Belkhatir, N., Ahmed-Nacer, M., 1995. Major issues on process software engineering environments. *Information Sciences* 83, 1–21. URL: <http://www.sciencedirect.com/science/article/pii/002002559400056H>, doi:[http://dx.doi.org/10.1016/0020-0255\(94\)00056-H](http://dx.doi.org/10.1016/0020-0255(94)00056-H).
- Belkhatir, N., Estublier, J., Melo, W., 1994. *Software Process Modelling and Technology*. Research Studies Press Ltd., Taunton, UK, UK. chapter ADELE-TEMPO: An Environment to Support Process Modelling and Enaction. pp. 187–222. URL: <http://dl.acm.org/citation.cfm?id=199318.199334>.

- Best, C., Storey, M.A., Michaud, J., 2002. Designing a component-based framework for visualization in software engineering and knowledge engineering, in: Proceedings of the 14th international conference on Software engineering and knowledge engineering, ACM, New York, NY, USA. pp. 323–326. URL: <http://doi.acm.org/10.1145/568760.568817>, doi:10.1145/568760.568817.
- Biehl, M., 2010. Supporting Model Evolution in Model-Driven Development of Automotive Embedded Systems. Licentiate thesis. KTH, Royal Institute of Technology.
- Biehl, M., 2011. Tool Integration Language (TIL). Technical Report 2011:14. KTH Royal Institute of Technology, Division of Mechatronics. URL: <http://www.md.kth.se/~biehl/files/papers/til.pdf>.
- Biehl, M., 2013. A Modeling Language for the Description and Development of Tool Chains for Embedded Systems. Ph.D. thesis. KTH Royal Institute of Technology.
- Biehl, M., DeJiu, C., Törngren, M., 2010a. Integrating safety analysis into the model-based development toolchain of automotive embedded systems, in: Proceedings of the ACM SIGPLAN/SIGBED 2010 conference on Languages, compilers, and tools for embedded systems, ACM, New York, NY, USA. pp. 125–132. URL: <http://doi.acm.org/10.1145/1755888.1755907>, doi:10.1145/1755888.1755907.
- Biehl, M., El-Khoury, J., Loiret, F., Törngren, M., 2011. A domain specific language for generating tool integration solutions, in: 4th Workshop on Model-Driven Tool & Process Integration (MDTPI2011) at the European Conference on Modelling Foundations and Applications (ECMFA 2011). URL: <http://www1.md.kth.se/~biehl/files/papers/mdtpi2011.pdf>.
- Biehl, M., El-Khoury, J., Loiret, F., Törngren, M., 2012a. On the modeling and generation of service-oriented tool chains. Software & Systems Modeling , 1–20URL: <http://dx.doi.org/10.1007/s10270-012-0275-7>, doi:10.1007/s10270-012-0275-7.
- Biehl, M., El-Khoury, J., Törngren, M., 2012b. High-level specification and code generation for service-oriented tool adapters, in: 2012 12th International Conference on Computational Science and Its Applications (ICCSA), pp. 35–42. doi:10.1109/ICCSA.2012.16.
- Biehl, M., Hong, J., Loiret, F., 2012c. Automated construction of data integration solutions for tool chains, in: Proceedings of the Seventh International Conference on Software Engineering Advances (ICSEA2012).
- Biehl, M., Sjöstedt, C.J., Törngren, M., 2010b. A modular tool integration approach - experiences from two case studies, in: Proceedings of the 3rd Workshop on Model-driven tool and Process Integration (MDTPI2010).

- Blessing, L.T., Chakrabarti, A., 2009. DRM, a Design Research Methodology. Springer-Verlag London Limited. doi:10.1007/978-1-84882-587-1.
- Bosua, R., Brinkkemper, S., 1995. Realisation of an integrated software engineering environment through heterogeneous CASE-tool integration, in: Software Engineering Environments [Conference], 1995., Proceedings, pp. 152–159. doi:10.1109/SEE.1995.393621.
- Brielmann, M., van den Broek, G., Höfner, G., Kupitz, E., Rudolph, S., Scheffström, D., Simmonds, I., 1993. Tool Integration - Environments and Frameworks. John Wiley & Sons Ltd.
- Brown, A.W., 1993a. Control integration through message-passing in a software development environment. *Software Engineering Journal* 8, 121–131.
- Brown, A.W., 1993b. An examination of the current state of IPSE technology, in: 15th International Conference on Software Engineering, 1993. Proceedings., pp. 338–347. doi:10.1109/ICSE.1993.346031.
- Brown, A.W., Carney, D.J., Morris, E.J., Smith, D.B., Zarrella, P.F., 1994. Principles of CASE Tool Integration. Oxford University Press.
- Brown, A.W., Feiler, P.H., 1992. An analysis technique for examining integration in a project support environment, in: Proceedings of the fifth ACM SIGSOFT symposium on Software development environments, ACM, New York, NY, USA. pp. 139–148. URL: <http://doi.acm.org/10.1145/142868.143764>, doi:10.1145/142868.143764.
- Brown, A.W., Feiler, P.H., Wallnau, K.C., 1992. Understanding integration in a software development environment, in: Proceedings of the Second International Conference on Systems Integration, 1992. ICSI '92., pp. 22–31. doi:10.1109/ICSI.1992.217286.
- Brown, A.W., McDermid, J.A., 1992. Learning from IPSE's mistakes. *IEEE Software* 9, 23–28. doi:10.1109/52.120598.
- Brown, A.W., Penedo, M.H., 1992. An annotated bibliography on integration in software engineering environments. *ACM SIGSOFT Software Engineering Notes* 17, 47–55. URL: <http://doi.acm.org/10.1145/140938.140944>, doi:10.1145/140938.140944.
- Brown, A.W., Penedo, M.H., 1994. "Environment integration" working group summary SETA2. *ACM Ada Letters* XIV, 85–92. URL: <http://doi.acm.org/10.1145/192839.192857>, doi:10.1145/192839.192857.
- Bruegge, B., Creighton, O., Helming, J., Kögel, M., 2008. Unicas - an ecosystem for unified software engineering, in: Proceedings of ICGSE '08: Distributed Software Development: Methods and Tools for Risk Management, pp. 12–17.

- Bull, R.I., 2008. Model Driven Visualization: Towards a Model Driven Engineering Approach for Information Visualization. Ph.D. thesis. Victoria, B.C., Canada, Canada.
- Buxton, J.N., 1980. STONEMAN, Requirements for Ada Programming Support Environments. Technical Report. Department of Defense.
- Carbone, J.N., Ekwaro-Osire, S., Tanik, M.M., 2010. A recombinant knowledge assimilation common process for recursive transdiscipline synthesis. *Journal of Integrated Design and Process Science* 14, 33–50.
- Ceccarelli, A., Vinerbi, L., Falai, L., Bondavalli, A., 2011. RACME: A framework to support V&V and certification, in: 2011 5th Latin-American Symposium on Dependable Computing (LADC), pp. 116–125. doi:10.1109/LADC.2011.21.
- Chen, M., Norman, R.J., 1992. A framework for integrated CASE. *IEEE Software* 9, 18–22.
- Chittister, C.G., Haimes, Y.Y., 1996. Systems integration via software risk management. *IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans* 26, 521–532. doi:10.1109/3468.531900.
- Clément, D., 1990. A distributed architecture for programming environments. *ACM SIGSOFT Software Engineering Notes* 15, 11–21. URL: <http://doi.acm.org/10.1145/99278.99280>, doi:10.1145/99278.99280.
- Conradi, R., Westfechtel, B., 1998. Version models for software configuration management. *ACM Computing Surveys (CSUR)* 30, 232–282. doi:10.1145/280277.280280.
- iFEST Consortium, 2013. industrial framework for embedded systems tools. URL: <http://www.artemis-ifest.eu/>.
- Crelier, O., Filho, R.S.S., Hasling, W.M., Budnik, C.J., 2011. Design principles for integration of model-driven quality assurance tools, in: Fifth Brazilian Symposium on Software Components, Architectures and Reuse (SBCARS), 2011, pp. 100–109. doi:10.1109/SBCARS.2011.11.
- Cuthill, B., 1994. Making sense of software engineering environment framework standards. *StandardView* 2, 188–202.
- Daberitz, D., Kelter, U., 1995. Rapid prototyping of graphical editors in an open SDE, in: Software Engineering Environments [Conference], 1995., Proceedings, pp. 61–71. doi:10.1109/SEE.1995.393610.
- Denzin, N.K., 2006. Sociological Methods, A Sourcebook. Transaction Publishers. chapter Part Twelve, Triangulation: A Case for Methodological and Combination Evaluation, Introduction. pp. 471–475.

- Dewar, R.G., 2005. Managing Software Engineering Artefact Metadata. Technical Report. Department of Computer Science, Heriot-Watt University.
- Earl, A., 1990. Principles of a reference model for computer aided software engineering environments, in: Long, F. (Ed.), *Software Engineering Environments*. Springer Berlin Heidelberg, volume 467 of *Lecture Notes in Computer Science*, pp. 115–129. URL: http://dx.doi.org/10.1007/3-540-53452-0_36, doi:10.1007/3-540-53452-0_36.
- Emmerich, W., 1996. Tool specification with GTSL, in: *Proceedings of the 8th International Workshop on Software Specification and Design, 1996.*, pp. 26–35. doi:10.1109/IWSSD.1996.501144.
- Endig, M., Jesko, D., 2001. Engineering processes - on an approach to realize a dynamic process control. *Journal of Integrated Design and Process Science* 5, 65–82. URL: <http://dl.acm.org/citation.cfm?id=1241721>.1241727.
- Ertürkmen, K.A., 2010. A Process Modeling Based Method for Identification and Implementation of Software Development Tool Integration-Tuples. Ph.D. thesis. The Middle East Technical University. Ankara.
- Eskeli, J., Maurolagoitia, J., Polcaro, C., 2011. PSW - a framework-based tool integration solution for global collaborative software development, in: *Proceedings of ICSEA 2011, The Sixth International Conference on Software Engineering Advances*, pp. 124–129.
- Fenton, N., Pfleeger, S.L., Glass, R.L., 1994. Science and substance: A challenge to software engineers. *IEEE Software* 11, 86–95. doi:10.1109/52.300094.
- Finkelstein, A., Gabbay, D., Hunter, A., Kramer, J., Nuseibeh, B., 1994. Inconsistency handling in multi-perspective specifications. *IEEE Transactions on Software Engineering* 20, 569–578. doi:10.1109/32.310667.
- Flyvbjerg, B., 2011. *The SAGE Handbook of Qualitative Research*. 4 ed.. SAGE Publications, Inc.. chapter 17. Case Study. pp. 301–316.
- Fowler, J., 1995. STEP for Data Management, Exchange and Sharing. *Technology Appraisals*.
- Gautier, B., Loftus, C., Sherratt, E., Thomas, L., 1995a. Tool integration: Experiences and directions, in: *ICSE 1995. 17th International Conference on Software Engineering, 1995.*, pp. 315–315.
- Gautier, R.J., Loftus, C.W., Sherratt, E.M., Thomas, L., 1995b. Tool integration: Experiences from the BOOST project, in: *Proceedings of the 1995 Software Engineering Environment Conference, IEEE Computer Society, Washington, DC, USA*. pp. 171–181. URL: <http://dl.acm.org/citation.cfm?id=829498>.829851, doi:10.1109/SEE.1995.393619.

- Giese, H., Neumann, S., Niggemann, O., Schätz, B., 2010. Model-based integration, in: Proceedings of the 2007 International Dagstuhl conference on Model-based engineering of embedded real-time systems, Springer-Verlag. pp. 17–54. URL: <http://dl.acm.org/citation.cfm?id=1927558.1927561>.
- Gimenes, I.M.S., Ciferri, C.D.A., Nanni, E.L., Sabião, S.B., 1997. An object-oriented method for designing sharable data schemas in software engineering environments. UNIMAR 4, 937–953.
- Glaser, B.G., Strauss, A.L., 1967. The Discovery of Grounded Theory, Strategies for Qualitative Research. AldineTransaction.
- Google, 2013. Google scholar. URL: <http://scholar.google.se/>.
- Grundy, J., Hosking, J., Zhu, N., Liu, N., 2006. Generating domain-specific visual language editors from high-level tool specifications, in: 21st IEEE/ACM International Conference on Automated Software Engineering, 2006. ASE '06., pp. 25–36. doi:10.1109/ASE.2006.39.
- Grundy, J., Mugridge, W., Hosking, J., 2000. Constructing component-based software engineering environments: Issues and experiences. Information and Software Technology 42, 103–114.
- Gu, W., 2012. Tool Integration: Model-Based Tool Adapter Construction and Discovery Conforming to OSLC. Master thesis. Uppsala University, Department of Information Technology.
- Haase, T., Nagl, M., 2011. Application integration within an integrated design environment. Computers and Chemical Engineering 35, 736–747. URL: <http://www.sciencedirect.com/science/article/pii/S0098135411000147>, doi:<http://dx.doi.org/10.1016/j.compchemeng.2011.01.012>.
- Harrison, W., Ossher, H., Kavianpour, M., 1992. Integrating coarse-grained and fine-grained tool integration, in: Proceedings of the Fifth International Workshop on Computer-Aided Software Engineering, 1992., pp. 23–35. doi:10.1109/CASE.1992.200127.
- Harvey, J.G., Marlin, C.D., 1997. Comparing inter-tool communication in control-centred tool integration frameworks, in: Eighth Conference on Software Engineering Environments, pp. 67–81. doi:10.1109/SEE.1997.591818.
- Hein, C., Ritter, T., Wagner, M., 2009. Model-driven tool integration with ModelBus, in: Proceedings of the 11th International Conference on Enterprise Information Systems.
- Herzog, E., Törne, A., 1999. A seed for a STEP application protocol for systems engineering, in: Proceedings of the 1999 IEEE Conference on Engineering of Computer-Based Systems, IEEE Computer Society, Washington, DC, USA. pp. 174–180. URL: <http://dl.acm.org/citation.cfm?id=1867775.1867798>.

- Holt, R.C., Schürr, A., Sim, S.E., Winter, A., 2006. Gxl: A graph-based standard exchange format for reengineering. *Science of Computer Programming* 60, 149–170. URL: <http://dx.doi.org/10.1016/j.scico.2005.10.003>, doi:10.1016/j.scico.2005.10.003.
- International Electrotechnical Commission, 2010. BS/IEC 61508:2010, functional safety of electrical/electronic/programmable electronic safety-related systems.
- International Organization for Standardization, 2011. ISO 26262:2011, road vehicles - functional safety.
- Ison, R., 1990. An experimental Ada programming support environment in the HP CASEdge integration framework, in: Long, F. (Ed.), *Software Engineering Environments*. Springer Berlin Heidelberg. volume 467 of *Lecture Notes in Computer Science*, pp. 179–193. URL: http://dx.doi.org/10.1007/3-540-53452-0_41, doi:10.1007/3-540-53452-0_41.
- Izza, S., 2009. Integration of industrial information systems: From syntactic to semantic integration approaches. *Enterp. Inf. Syst.* 3, 1–57. URL: <http://dx.doi.org/10.1080/17517570802521163>, doi:10.1080/17517570802521163.
- Jelschen, J., Winter, A., 2011. Towards a catalogue of software evolution services. *Softwaretechnik Trends* 31.
- Jucovschi, C., 2012. Cost-effective integration of MKM semantic services into editing environments, in: Jeuring, J., Campbell, J.A., Carette, J., Reis, G., Sojka, P., Wenzel, M., Sorge, V. (Eds.), *Intelligent Computer Mathematics*. Springer Berlin Heidelberg. volume 7362 of *Lecture Notes in Computer Science*, pp. 96–110. URL: http://dx.doi.org/10.1007/978-3-642-31374-5_7, doi:10.1007/978-3-642-31374-5_7.
- Kapsammer, E., Reiter, T., Schwinger, W., 2006. Model-based tool integration - state of the art and future perspectives, in: In the Proceedings of the 3rd International Conference on Cybernetics and Information Technologies, Systems and Applications.
- Kienle, H.M., Müller, H.A., 2007. A lightweight taxonomy to characterize component-based systems, in: Sixth International IEEE Conference on Commercial-off-the-Shelf (COTS)-Based Software Systems, 2007. ICCBSS '07., pp. 192–204. doi:10.1109/ICCBSS.2007.2.
- Kordon, F., Mounier, J.L., 1997. FrameKit and the prototyping of CASE environments, in: 8th IEEE International Workshop on Rapid System Prototyping, 1997. Shortening the Path from Specification to Prototype. Proceedings., pp. 91–97. doi:10.1109/IWRSP.1997.618846.
- Kothari, C., 2004. *Research Methodology, Methods and Techniques*. New Age International (P) Ltd., Publishers.

- Kouamou, G.E., 2011. New Fundamental Technologies in Data Mining. InTech. chapter 13. A Software Architecture for Data Mining Environment. pp. 241–258. URL: <http://www.intechopen.com/books/new-fundamental-technologies-in-data-mining/a-software-architecture-for-data-mining-environment>, doi:10.5772/13351.
- Koudri, A., Petrisor, T., Champeau, J., Leilde, V., 2011. A framework for integrating ESL tools, in: Proceedings of the First Workshop on Process-based approaches for Model-Driven Engineering (PMDE).
- Kramler, G., Kappel, G., Reiter, T., Kapsammer, E., Retschitzegger, W., Schwinger, W., 2006. Towards a semantic infrastructure supporting model-based tool integration, in: Proceedings of the 2006 international workshop on Global integrated model management, ACM, New York, NY, USA. pp. 43–46. URL: <http://doi.acm.org/10.1145/1138304.1138314>, doi:10.1145/1138304.1138314.
- Kääriäinen, J., 2011. Towards an Application Lifecycle Management Framework. Ph.D. thesis. VTT.
- Kurniawan, D., Abramson, D., 2008. An IDE framework for grid application development, in: 9th IEEE/ACM International Conference on Grid Computing, 2008., pp. 184–191. doi:10.1109/GRID.2008.4662798.
- Kurniawan, D., Abramson, D., 2011. ISENGARD: An infrastructure for supporting e-science and grid application development. Concurrency and Computation: Practice and Experience 23, 390–414. URL: <http://dx.doi.org/10.1002/cpe.1662>, doi:10.1002/cpe.1662.
- Long, F.W., Morris, E.J., 1993. An Overview of PCTE: A Basis for a Portable Common Tool Environment. Technical Report 218, CMU/SEI-93-TR-1, ESC-TR-93-175. Software Engineering Institute. URL: <http://repository.cmu.edu/sei/218>.
- Losavio, F., Ortega, D., Pérez, M., 2002. Modeling EAI, in: Proceedings of the XII International Conference of the Chilean Computer Science Society, IEEE Computer Society, Washington, DC, USA. pp. 195–203. URL: <http://dl.acm.org/citation.cfm?id=829496.829731>.
- Loutchkina, I., Nesterov, S., 2010. Exploring the factors contributing to system integration risks. International Journal of Intelligent Defence Support Systems 3, 325–354. URL: <http://inderscience.metapress.com/content/4656085QJ1M06722>, doi:10.1504/IJIDSS.2010.037096.
- Maalej, W., 2009. Task-first or context-first? tool integration revisited, in: 24th IEEE/ACM International Conference on Automated Software Engineering, 2009. ASE '09., pp. 344–355. doi:10.1109/ASE.2009.36.

- Mampilly, T., Ramnath, R., Irani, S., 2005. PFAST: An eclipse-based integrated tool workbench for facilities design, in: Proceedings of the 2005 OOPSLA workshop on Eclipse technology eXchange, ACM, New York, NY, USA. pp. 80–84. URL: <http://doi.acm.org/10.1145/1117696.1117713>, doi:10.1145/1117696.1117713.
- Marquardt, W., Nagl, M., 2004. Workflow and information centered support of design processes - the improve perspective. Computers and Chemical Engineering 29, 65–82. doi:10.1016/j.compchemeng.2004.07.018.
- Miatidis, M., 2007. Integrated Experience-Based Support of Cooperative Engineering Design Processes. Ph.D. thesis. Rheinisch-Westfälischen Technischen Hochschule Aachen.
- Michaud, J., Storey, M.A., Müller, H., 2001. Integrating information sources for visualizing Java programs, in: IEEE International Conference on Software Maintenance, 2001. Proceedings., pp. 250–258. doi:10.1109/ICSM.2001.972738.
- Michaud, J.W., 2003. A Software Customization Framework. Master thesis. University of Victoria.
- Nakagawa, E.Y., Ferrari, F.C., Sasaki, M.M., Maldonado, J.C., 2011. An aspect-oriented reference architecture for software engineering environments. Journal of Systems and Software 84, 1670–1684. URL: <http://www.sciencedirect.com/science/article/pii/S0164121211001038>, doi:<http://dx.doi.org/10.1016/j.jss.2011.04.052>.
- Nuseibeh, B., Kramer, J., Finkelstein, A., 1994. A framework for expressing the relationships between multiple views in requirements specification. IEEE Transactions on Software Engineering 20, 760–773. doi:10.1109/32.328995.
- Nuseibeh, B.A., 1994. A Multi-Perspective Framework for Method Integration. Ph.D. thesis. Imperial College of Science, Technology and Medicine, University of London, Department of Computing.
- Osterweil, L., 1988. A Process-Object Centered View of Software Environment Architecture. Technical Report CU-CS-332-86. University of Colorado at Boulder, Department of Computer Science.
- Pesola, J.P., Tanner, H., Eskeli, J., Parviainen, P., Bendas, D., 2011. Integrating early V&V support to a GSE tool integration platform, in: 2011 Sixth IEEE International Conference on Global Software Engineering Workshop (ICGSEW), pp. 95–101. doi:10.1109/ICGSE-W.2011.32.
- Pohl, K., Weidenhaupt, K., 1997. A contextual approach for process-integrated tools. SIGSOFT Software Engineering Notes 22, 176–192. URL: <http://doi.acm.org/10.1145/267896.267910>, doi:10.1145/267896.267910.

- Pohl, K., Weidenhaupt, K., Dömges, R., Haumer, P., Jarke, M., Klamma, R., 1999. PRIME - toward process-integrated modeling environments. *ACM Transactions on Software Engineering and Methodology* 8, 343–410. URL: <http://doi.acm.org/10.1145/322993.322995>, doi:10.1145/322993.322995.
- Pratt, M.J., 2005. ISO 10303, the STEP standard for product data exchange, and its PLM capabilities. *International Journal of Product Lifecycle Management* 1, 86–94. doi:10.1504/IJPLM.2005.007347.
- Qureshi, T.N., 2012. Enhancing Model-Based Development of Embedded Systems, Architecture-Centric Modeling, Simulation and Model-Transformation in an Automotive Context. Ph.D. thesis. KTH Royal Institute of Technology.
- Reiss, S.P., 1999. The Desert environment. *ACM Transactions on Software Engineering and Methodology* 8, 297–342. URL: <http://doi.acm.org/10.1145/322993.322994>, doi:10.1145/322993.322994.
- Reynolds, C.W., 1987. Flocks, herds and schools: A distributed behavioral model. *SIGGRAPH Computer Graphics* 21, 25–34. URL: <http://doi.acm.org/10.1145/37402.37406>, doi:10.1145/37402.37406.
- Schneider, R., Marquardt, W., 2002. Information technology support in the chemical process design life cycle. *Chemical Engineering Science* 57, 1763 – 1792. URL: <http://www.sciencedirect.com/science/article/pii/S0009250902000751>, doi:10.1016/S0009-2509(02)00075-1.
- Shi, J., 2007. Model and Tool Integration in High Level Design of Embedded Systems. Ph.D. thesis. KTH Royal Institute of Technology.
- Skoglund, D., 2012. A Standardized Approach to Tool Integration. Technical Report 12 019. Uppsala University, Department of Information Technology.
- Sliski, T.J., Billmers, M.P., Clarke, L.A., Osterweil, L.J., 2001. An architecture for flexible, evolvable process-driven user-guidance environments, in: *Proceedings of the 8th European software engineering conference held jointly with 9th ACM SIGSOFT international symposium on Foundations of software engineering*, ACM, New York, NY, USA. pp. 33–43. URL: <http://doi.acm.org/10.1145/503209.503214>, doi:10.1145/503209.503214.
- Special Committee 205 of RTCA, Inc., 2011a. DO-178C, software considerations in airborne systems and equipment certification.
- Special Committee 205 of RTCA, Inc., 2011b. DO-330, software tool qualification considerations.
- Stavridou, V., 1999. Integration in software intensive systems. *Journal of Systems and Software* 48, 91–104. URL: [http://dx.doi.org/10.1016/S0164-1212\(99\)00049-7](http://dx.doi.org/10.1016/S0164-1212(99)00049-7), doi:10.1016/S0164-1212(99)00049-7.

- Stoeckle, H., Grundy, J., Hosking, J., 2005. A framework for visual notation exchange. *Journal of Visual Languages and Computing* 16, 187–212. URL: <http://dx.doi.org/10.1016/j.jvlc.2004.08.002>, doi:10.1016/j.jvlc.2004.08.002.
- Sum, S., Ibold, C., 1994. Information technology support for concurrent and simultaneous engineering - tool integration in a meta framework, in: *Proceedings of TENCON '94. IEEE Region 10's Ninth Annual International Conference. Theme: Frontiers of Computer Technology. 1994.*, pp. 1068–1073. doi:10.1109/TENCON.1994.369155.
- Tashakkori, A., Teddlie, C., 1998. *Mixed Methodology, Combining Qualitative and Quantitative Approaches*. Sage Publications.
- Thomas, I., Nejme, B.A., 1992. Definitions of tool integration for environments. *IEEE Software* 9, 29–35.
- Tilley, S., Gerdes, J., Hamilton, T., Huang, S., Müller, H., Smith, D., Wong, K., 2004. On the business value and technical challenges of adopting web services. *Journal of Software Maintenance and Evolution: Research and Practice* 16, 31–50. URL: <http://dx.doi.org/10.1002/smr.283>, doi:10.1002/smr.283.
- Tilley, S.R., 1994. Domain-retargetable reverse engineering ii: Personalized user interfaces, in: *Proceedings of the International Conference on Software Maintenance, 1994.*, pp. 336–342. doi:10.1109/ICSM.1994.336761.
- Tilley, S.R., 1995. *Domain-Retargetable Reverse Engineering*. Ph.D. thesis. University of Victoria, Department of Computer Science. Victoria, B.C., Canada, Canada.
- Tilley, S.R., 2000. The canonical activities of reverse engineering. *Annals of Software Engineering* 9, 249–271. URL: <http://dx.doi.org/10.1023/A%3A1018985010508>, doi:10.1023/A:1018985010508.
- Tilley, S.R., Paul, S., Smith, D.B., 1996. Towards a framework for program understanding, in: *Fourth Workshop on Program Comprehension, 1996, Proceedings.*, pp. 19–28. doi:10.1109/WPC.1996.501117.
- Tilley, S.R., Smith, D.B., 1996. *Coming Attractions in Program Understanding*. Technical Report CMU/SEI-96-TR-019. Software Engineering Institute. Carnegie Mellon University, Pittsburgh, Pennsylvania. URL: <http://www.sei.cmu.edu/library/abstracts/reports/96tr019.cfm>.
- Valetto, G., Kaiser, G.E., 1996. Enveloping sophisticated tools into process-centered environments. *Automated Software Engineering* 3, 309–345. URL: <http://dx.doi.org/10.1007/BF00132571>, doi:10.1007/BF00132571.
- Vasista, T., AlSudairi, M.A., 2012. Web portal integration architecture approaches. *International Journal on Web Service Computing* 3.

- Vasista, T., AlSudairi, M.A., 2013. Service-oriented architecture (SOA) and semantic web services for web portal integration, in: Meghanathan, N., Nagamalai, D., Chaki, N. (Eds.), *Advances in Computing and Information Technology*. Springer Berlin Heidelberg, volume 177 of *Advances in Intelligent Systems and Computing*, pp. 253–261. URL: http://dx.doi.org/10.1007/978-3-642-31552-7_27, doi:10.1007/978-3-642-31552-7_27.
- Wallace, E., Wallnau, K.C., 1996. A situated evaluation of the object management group’s (OMG) object management architecture (OMA), in: *Proceedings of the 11th ACM SIGPLAN conference on Object-oriented programming, systems, languages, and applications*, ACM, New York, NY, USA, pp. 168–178. URL: <http://doi.acm.org/10.1145/236337.236353>, doi:10.1145/236337.236353.
- Wasserman, A.I., 1990. Tool integration in software engineering environments, in: *Software Engineering Environments: International Workshop on Environments*, Chinon, France, September 1989, *Proceedings*.
- Wasserman, A.I., 1995. Requirements for OO design environments, in: *Proceedings of the 1995 Software Engineering Environment Conference*, IEEE Computer Society, Washington, DC, USA, pp. 207–211. URL: <http://dl.acm.org/citation.cfm?id=829498.829847>, doi:10.1109/SEE.1995.393616.
- Wasserman, A.I., 1996. Toward a discipline of software engineering. *IEEE Software* 13, 23–31. doi:10.1109/52.542291.
- Welsh, J., Yang, Y., 1994. *Integration of Semantic Tools into Document Editors*. Technical Report 94-2. Software Verification Research Centre, Department of Computer Science, The University of Queensland.
- Wende, C., Assmann, U., Zivkovic, S., Kühn, H., 2011. Feature-based customisation of tool environments for model-driven software development, in: *2011 15th International Software Product Line Conference (SPLC)*, pp. 45–54. doi:10.1109/SPLC.2011.29.
- Wende, C., Zivkovic, S., Assmann, U., Kühn, H., 2010. *Feature-Based Customisation of MDSD Tool Environments*. Technical Report TUD-FI10-05-Juli. Technische Universität Dresden, Fakultät Informatik.
- Westfechtel, B., 1999. *Models and Tools for Managing Development Processes*. Springer-Verlag, Berlin, Heidelberg.
- Wicks, M., 2004. Tool integration in software engineering: The state of the art. *Integration, The VLSI Journal* August, 1–26. doi:10.1.1.59.1969.
- Wicks, M., Dewar, R., 2007. A new research agenda for tool integration. *The Journal of Systems and Software* 80, 1569–1585.
- Wong, K., 1999. *The Reverse Engineering Notebook*. Ph.D. thesis. University of Victoria, Department of Computer Science. Victoria, B.C., Canada, Canada.

- Woodside, C.M., 1995. A three-view model for performance engineering of concurrent software. *IEEE Transactions on Software Engineering* 21, 754–767. URL: <http://dx.doi.org/10.1109/32.464545>, doi:10.1109/32.464545.
- Yazbek, H., 2010. Framework of Quality Measurement in CASE Tool Based Software Development. Ph.D. thesis. Otto von Guericke Universität Magdeburg.
- Zelkowitz, M.V., 1993. Use of an environment classification model, in: *ICSE '93: Proceedings of the 15th international conference on Software Engineering*, pp. 348–357. doi:10.1109/ICSE.1993.346030.
- Zelkowitz, M.V., 1996. Modeling software engineering environment capabilities. *Journal of Systems and Software* 35, 3–14. URL: <http://www.sciencedirect.com/science/article/pii/0164121295000828>, doi:[http://dx.doi.org/10.1016/0164-1212\(95\)00082-8](http://dx.doi.org/10.1016/0164-1212(95)00082-8).
- Zhang, W., Møller-Pedersen, B., Hansen, K.T., 2011. Metamodel-based tool integration, in: *Norsk Informatikkonferanse NIK 2011*, Tapir Akademisk Forlag. pp. 251–254.