Res Eng Design (2017) 28:153–184
DOI 10.1007/s00163-016-0226-3

ORIGINAL PAPER

Perspectives on iteration in design and development

David C. Wynn[1] · Claudia M. Eckert[2]

Received: 24 October 2015 / Revised: 15 March 2016 / Accepted: 18 March 2016 / Published online: 5 April 2016

Abstract Design, development, and other projects inevitably involve iteration. Iteration has positive effects, such as enabling progressive generation of knowledge, enabling concurrency, and integrating necessary changes, but it also increases duration and cost of a project. Managing iteration is thus an important issue in practice, but can be challenging due in part to a profusion of issues and terminologies. This article contributes a literature summary and integrating taxonomy to clarify the different perspectives on iteration. It brings together insights into iteration gained from different research communities (mainly design and product development, alongside selected work in construction management and software project management) and different research approaches (including conceptual frameworks, mathematical and simulation models, case studies and surveys, and protocol studies). By differentiating the issues and providing a uniform terminology, the article maps insights developed to date and may help situate future analyses of iterative processes.

Keywords Iteration · Design and development · Literature review · Integrating taxonomy

✉ David C. Wynn
d.wynn@auckland.ac.nz

1 Department of Mechanical Engineering, The University of Auckland, 20 Symonds Street, Auckland 1010, New Zealand

2 Department of Engineering and Innovation, The Open University, Walton Hall, Milton Keynes MK7 6AA, UK

1 Introduction

Iteration is a fact of life in any project. The larger, more novel and more interconnected a project, the more of an issue it can be (Mihm et al. 2003; Braha and Bar-Yam 2007). Iteration is thus especially prominent in the development of complex systems such as aircraft—in this context, even world-leading companies expect it to be inevitable in their development projects. Iteration is difficult to manage and control, and is commonly associated with project overruns.

The importance of iteration is well recognised in design and product development research, as well as other disciplines such as software and construction. According to one PhD dissertation on the topic, “articles on engineering design frequently allude to its iterative character in literally the first paragraph” (Safoutin 2003). Empirical studies highlight the ubiquity of iteration in real-world development projects (e.g. Osborne 1993), while a 2011 survey of practitioners and academics suggested that experts believe “iterative” is amongst the most important characteristics of the design process (Maier and Störrle 2011). Yassine and Braha (2003) write that iteration is one of four critical problems in the management of concurrent engineering and is inevitable because: (1) designers cannot consider all issues impinging on a design decision at once; (2) it is usually not possible to compute a design directly from a set of requirements; and (3) unpredictable interactions often occur such that a design does not perform in the expected way and must be adjusted (Yassine and Braha 2003). Another key cause of iteration is the decomposition of development work and its dispersion amongst teams, leading to iteration when problems are discovered during integration (Eppinger et al. 1994). Iteration can also have positive effects including exploration of concepts, finding and correcting flaws, and enabling development under complexity, uncertainty and change (Le 2012). Overall, the accepted view, as expressed by Smith and Eppinger (1997a), is that understanding iteration is fundamental to improving and accelerating product development, as well as projects in other domains.

Reflecting this agreed importance, many researchers have considered the issues surrounding iteration. A number of distinct perspectives have been developed that are embodied in different models of the iteration process. Stage-based prescriptive models of product development (PD) show feedback loops from a late project stage to an earlier one (Pahl and Beitz 1996). This feedback can manifest between stages in the development process, from in-service to the development process, and back to the finding of markets for new products (Kline 1985). It can drive iteration within a project, as well as generational learning across consecutive or concurrent projects (Eppinger et al. 1994). Models of design cognition describe a fundamentally iterative process of exploration and convergence, in which alternatives are repeatedly generated, analysed, and evaluated (Wynn and Clarkson 2005). These cycles of problem-solving, which occur when an obvious solution is not apparent for a particular issue, can involve a single individual and can also occur on a broader scale, e.g. involving a whole team or department (Clark et al. 1987). A substantial body of research presents iteration as a consequence of complex interdependencies in a process, project, or system architecture (e.g. Eppinger 1991; Smith and Eppinger 1997a). Numerous tools and approaches have also been developed to mitigate, manage, and exploit iteration.

Overall, many researchers agree that design processes and large projects such as product development and construction are iterative in nature. The importance of managing iteration is also well established in the research literature. However, there are many issues and perspectives on the causes, effects, and types of iteration. It is often referred to using different terms, such as rework, loop-backs, feedback loops, and churn and has been described as revisiting either tasks, or problems, or parts of a design. Researchers also consider processes from different perspectives and focus their descriptions of iteration on those issues pertinent to their objectives. Although the different views may be considered “complementary rather than competing” (Smith and Tjandra 1998), the authors of the present article propose that a conceptually integrated approach could be key to more effectively managing iteration. In particular, most real-world situations emphasise only a small subset of the issues that researchers have discussed (Wynn 2007). An approach to crisply delineate the properties of an iterative situation would therefore help to pinpoint the most relevant issues and insights. The present article contributes towards this goal by (1) compiling a structured overview of issues surrounding iteration; and (2) drawing on the overview to develop an integrating taxonomy for describing and differentiating between iterative situations. It is hoped that these contributions may suggest directions for future research as well as help researchers position their work in relation to state of the art.

Because substantial research has been done on iteration in design and development, but with limited conceptual integration, the summary of issues and taxonomy was developed through *integrative literature review*. This is “a form of research that reviews, critiques, and synthesises representative literature on a topic in an integrated way such that new frameworks and perspectives on the topic are generated” (Torraco 2005). A key challenge in this case was that, although iteration is an important element of many publications relating to design and development, many authors do not explicitly frame their contribution in terms of insights into iteration. Thus, it was necessary to study publications in depth to identify relevant key concepts, grasp the relationships between them, and synthesise a framework to describe selected features of this landscape in a manner that yields additional insight.

Ultimately, the article provides a perspective on the literature that is influenced by the research process that was followed as well as the authors’ decisions regarding what to include and what to emphasise. These three influences are discussed in the next three paragraphs.

First, in terms of process, a list of influential articles was initially compiled based on the authors’ prior knowledge of the research area. Bibliographies from these publications were then used to identify additional relevant sources, including conference papers, books, and dissertations. These were digested and their bibliographies studied in turn. Internet search was also used to identify relevant articles as issues were identified. Peer reviewers suggested additional sources. As each publication was considered, a framework to organise the perspectives was synthesised and incrementally adjusted. This process continued until further reading ceased to yield significant changes to the emerging manuscript. Once the manuscript was completed, original sources listed in the bibliography were revisited to verify the review’s accuracy. Some adjustments were subsequently made.

Second, to align with the journal, the scope of the review was originally set on design and product development. Through the process explained above, this initial scope was expanded to include selected work from other development domains where significant insights into iteration were found, in particular construction management and software process management. Due to space limitations, we focus on literature that develops insights into iteration in an empirical or theoretical setting, and do not review prescriptive methods and tools intended to support practice (some pointers for further reading on this topic are provided in Sect. 5.6). For the same reason, we focus mainly on iteration within the context of a single project or programme, and on iteration which involves design activity. For instance, reorientation of parts during assembly is not explicitly considered, although it could arguably be perceived as a form of iteration that might be reduced by appropriate design or design changes (Boothroyd 1994; Braha and Maimon 1998, 2013).

Due to the vast number of publications that touch upon iteration even within the focus areas, an emphasis was placed on identifying influential articles to support key points, rather than exhaustively reviewing all developments of each idea. To indicate the sources of the selected literature, the five journals that appear most frequently in this article's bibliography are as follows: *Management Science*, *Research in Engineering Design*, *Design Studies*, *Journal of Mechanical Design*, and *IEEE Transactions on Engineering Management*.

Third, the emphasis and organisation of the review emerged during the process explained above, as key features of the literature were identified and digested. Our interpretation of the literature was guided by our previous industry case studies (of which some are summarised in Eckert and Clarkson 2010). A number of these focused on creating support for complex design considering its iterative characteristics (e.g. Flanagan et al. 2007; Wynn et al. 2014). We also drew on the second author's empirical research into engineering change (e.g. Eckert et al. 2004), which is closely related to iteration.

2 Review of insights into iteration

This section provides a structured review of insights into iteration gained from the literature. The objective of the section is (1) to highlight key insights and provide an organised source of reference; and (2) to convey different iterative situations that have been described in the literature. This provides the basis for the integrating taxonomy developed in later sections.

Through the process explained in Sect. 1, it was determined that insights into iteration may be organised into two categories. The first concerns iteration on the micro-level, focusing on the iterative characteristics of individual or small-group design activity and the creative or problem-solving episodes that happen throughout a project. The second category of insights concerns iteration on the macro-level, which refers to large-scale iterative processes in the project context. Research taking the former perspective has focused on issues such as design cognition and creativity, while in the latter perspective iteration related to

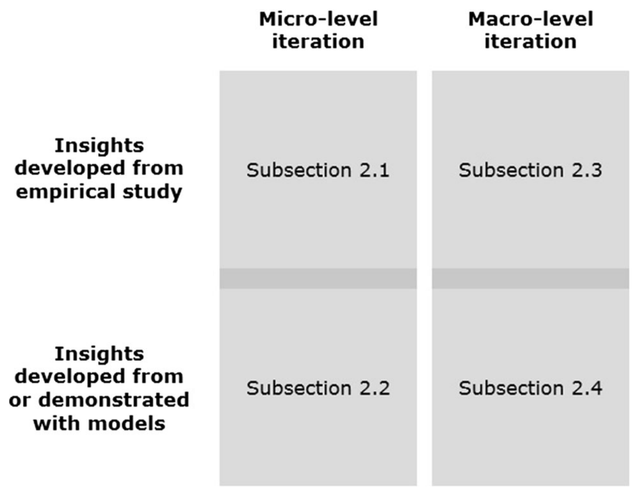

Fig. 1 An overview of Sect. 2

interdependency, concurrency, and coordination has received substantial attention. On both micro- and macro-levels, research work can be further divided into two subcategories. Publications in the first subcategory focus mainly on developing or demonstrating insights by empirical study. Publications in the second subcategory focus mainly on developing detailed models of the iteration process. Although some publications contribute to both subcategories and there is thus a certain amount of overlap between them, they are nevertheless useful to organise the discussion.

These categories and subcategories may be visualised as a grid as shown in Fig. 1. Within each cell of the grid, authors have developed insights relating to iteration as summarised in Table 1. The next four subsections discuss each cell of the grid in turn.

2.1 Empirical insights into micro-level iteration

Researchers have developed insights into iteration by undertaking observations and laboratory experiments in which participants (often university students) are asked to solve a simple design problem in a time spanning no more than a few hours. These studies provide insight into iteration on the micro-level, such as cognitive iterations within a designer's mind or iterations in a small collaborative setting.

A recurring theme in this literature has been to examine how iteration allows progress to be made in a design context that is *ill-defined* and *ambiguous*, via a process having characteristics such as *goal-directed*, *situated*, *responsive*, *opportunistic*, and *creative*. For instance, Adams and Atman (2000) write that design iterations indicate "a progression through levels of understanding as the designer

Table 1 Summary of insights relating to iteration with selected supporting references

ES	Mo	Insight
Micro-level iteration...		
2.1		...is used/is necessary to elaborate ill-structured problems (Simon 1973)
2.1	2.2	...is used/is necessary to develop problem and solution space concurrently (Schön and Wiggins 1992)
2.1.1		...is used/is necessary in non-routine design (Guindon 1990)
2.1.2		...is used by experienced designers to explore concepts; by novices to improve them (Smith and Tjandra 1998)
2.1.3		...over concepts leads to better designs, but iteration over details may not (Smith and Tjandra 1998)
2.1.4		...can be reduced by structuring a process to solve coupled problems in groups (Smith et al. 1992)
2.1.4		...is most effective if goals are clarified beforehand (Badke-Schaub and Gehrlicher 2003)
2.1.5		...may help to monitor progress and navigate the design process (Adams and Atman 2000)
	2.2	...is needed to solve complex problems (Moen and Norman 2010)
	2.2	...may help to deal with a changing context (Moen and Norman 2010)
Macro-level iteration...		
2.3.1		...helps to deal with a changing context or unclear situation (Cusumano 1997)
2.3.2		...has no universal positive or negative relationship to project performance (Dybå and Dingsøyr 2008)
2.3.3		...increases if testing is used earlier in PD, but this may reduce PD time overall (Thomke 1998)
2.3.4	2.4.1	...is increased by using prelim. info. but this may reduce PD time overall (Eastman 1980)
2.3.4	2.4.1	...is influenced by the characteristics of prelim. info, where used (Terwiesch et al. 2002; Krishnan et al. 1995)
2.3.5		...is increased by small design margins and high integration levels (Eckert et al. 2004)
2.3.5		...may be reduced by modularity, flexibility (Thomke 1997) and standardisation (Liker and Morgan 2006)
2.3.6		...is influenced by a complex network of factors acting simultaneously (Love and Edwards 2004)
2.3.6		...is influenced by the right-skewed degree distribution of PD networks (Braha and Bar-Yam 2004a, b)
2.3.6		...may be mitigated by disassortative structures arising from PD task properties (Braha et al. 2013)
	2.4.1	...increases with task sensitivity to changes in preliminary information (Krishnan et al. 1995)
	2.4.1	...caused by prelim. info. may be reduced by interaction between engineering functions (Bhuiyan et al. 2004)
	2.4.1	...increases as concurrency increases (AitSahlia et al. 1995)
	2.4.2	...is influenced by design review timing (Ha and Porteus 1995)
	2.4.3	...is influenced by the sequence of freezing design decisions or parameters (Krishnan et al. 1997b)
	2.4.4	...increases when solving larger and more strongly connected problems (Loch et al. 2003; Yassine et al. 2003)
	2.4.4	...may not converge if problem complexity exceeds a project's problem-solving capacity (Loch et al. 2003)
	2.4.4	...is increased by strong pairwise interactions between components being designed (Yassine et al. 2003)
	2.4.4	...may be reduced by a focus on central information-consuming/generating nodes (Braha and Bar-Yam 2007)
	2.4.4	...is strongly influenced by small changes in task time if close to capacity (Huberman and Wilkinson 2005)
	2.4.5	...may be reduced by increasing coordination intensity (Loch and Terwiesch 1998)
	2.4.5	...is increased by delays in coordination (Yassine et al. 2003)
	2.4.5	...increases when coordination needs exceed communication capacity in a project (Levitt et al. 1999)
	2.4.5	...can be dramatically reduced by accepting slightly lower design performance (Mihm et al. 2003)

ES = Insight developed from empirical study, Mo = Insight developed from or elaborated using a model. Numbers in these columns refer to subsections in which the corresponding insights are discussed

discovers and responds to new information about a problem or solution". Dorst and Cross (2001) describe this as a fundamentally iterative coevolution of the problem and solution spaces: only by generating solutions can the designer understand the problem they are trying to resolve with the design, because proposed solutions highlight the aspects of the problem that need to be considered (Kolodner and Wills 1996). This is needed because design problems are typically ill-structured at the outset (Simon 1973). Furthermore, the intermediate representations of a design, and in particular the ambiguous nature of many early design representations, have been said to inspire designers to see alternative solutions (Schön and Wiggins 1992). This again implies a fundamentally iterative process influenced by experience and imagination, such that different routes might emerge on different occasions (Braha and Maimon 1998). There may be no clear stopping criterion for early iterations (Adams and Atman 2000), partly because problem and solution are developed concurrently such that goals may be unclear at the outset, as noted above, and partly because early in the design process there may not be enough information about a design to confidently assess its performance.

Some empirical studies focus on unpacking the structure of the micro-level iterative process. For instance, Austin et al. (2001) report on a workshop in which teams of five designers completed a simplified building design problem. They find evidence for a dynamic described as whirling, in which designers rapidly iterate amongst several consecutive phases to complete the first of those phases, before iterating around the second and later phases to complete the second phase, and so forth. To give another example, Boudouh et al. (2006) study recordings of engineering students collaborating on a conceptual design process and find that in this case, the majority of iterations were expected (rather than unexpected), short in duration (rather than long), and value-adding (rather than correcting errors).

Such studies provide some insight into micro-level iterative dynamics, although it is not clear how far they can be generalised beyond the particular situations studied. Other empirical work indicates the contingent nature of the micro-level iteration process. The main insights extracted after reviewing such studies are discussed in the next subsections.

2.1.1 *Iteration is common in non-routine tasks*

Empirical studies have highlighted that iteration is especially necessary in non-routine or ambiguous situations. For instance, Guindon (1990) describes a think-aloud study of software professionals designing a lift control system, in which the design episodes are coded as relating to the domains of either problem, requirements, or solution on one of three levels of abstraction. Analysing the results, he characterises opportunistic progress as a fundamental feature of the design process, in which the designer skips back and forth (i.e. iterates) between domains and levels of abstraction. He argues that such deviations from a sequential process are needed to deal with the ill-structured nature of problem and requirements in most real design situations, and that a linear, top-down process is a special case that occurs only when a suitable solution decomposition is already known. Similarly, but in the context of mechanical design, Jin and Chusilp (2006) study the effect of whether a problem is creative or routine and whether it is constrained or unconstrained, on how designers iterate during conceptual design. They conclude that, in comparison with "constrained" design, "creative" design involves more frequent iterations amongst the activities of problem analysis, idea generation, composition of ideas from other ideas, and evaluation, and less time is spent on each iteration.

2.1.2 *Iteration is influenced by experience level and problem-specific knowledge*

The way designers iterate has been shown in several studies to depend on their level of experience. For instance, Ahmed et al. (2003) analyse a think-aloud protocol of aerospace designers engaged in real design tasks. They report that novice designers followed a simple iterative strategy of generate idea–implement idea–evaluate result (in the context they studied, the implementation step required time-consuming computational analysis). In contrast, they found that more experienced designers drew on their expertise to evaluate proposed ideas more thoroughly using a number of so-called design strategies before committing to implement them, resulting in less time and fewer cycles overall. Similarly, the experimental study of Smith and Leong (1998) found that experienced professionals generated and discarded multiple prototypes to learn about the design problem, while novices tended to "iterate their way to a solution" from a single initial concept. This is aligned with Atman et al. (1999)'s empirical conclusion that while iteration leads to better-quality designs, the relationship is stronger for freshman students than for seniors.

The structure of iterations has been shown not only to depend on designer experience, but also to evolve during the process as knowledge about the problem at hand increases and the nature of that problem accordingly changes. For instance, Smith and Tjandra (1998) concluded from another experiment that as iterations progress, synthesis tasks are completed more and more rapidly while analysis tasks become more time-consuming. They suggest this is because participants gradually develop deeper understanding of what changes may be technically feasible, justifying more intensive analysis.

2.1.3 *Iteration over concepts may lead to better designs*

Some empirical studies suggest that spending more effort to iteratively explore the space of design concepts yields better designs overall. For example, Chusilp and Jin (2006) undertake an experiment and find that more iterations and more time spent on each iteration typically lead to better quality and more novel concepts. Atman et al. (1999) code recordings of students working on an architectural design problem into transitions between three activities—problem scoping; developing alternative solutions; and project realisation—and similarly find that the number of transitions (implying iterations) between these steps is correlated to improved design performance. Smith and Tjandra (1998) also find that revisiting concepts led to higher quality. Notably, their study showed no significant correlation between number of detail design iterations and design performance. This is in agreement with the generally accepted idea that early decisions can disproportionately influence a design's eventual performance (e.g. Wyatt et al. 2012).

2.1.4 *Iteration may be positively influenced by managing the micro-structure of the design process*

Designers can direct their own design processes, and a number of studies indicate that the design process structure, or design strategies that are used, may influence the dynamics and value of iterations. For instance, Smith et al. (1992) study groups of students working together on an abstract design game, concluding that structuring the process so that strongly coupled subproblems are solved individually then combined reduces large-scope iterations and thus overall design time without impacting solution quality. Also considering process structure, Safoutin (2003) examines the behaviour of students faced with a parametric design task. He argues that "focused" iteration of one parameter based on feedback around changes to its value, before moving onto the next parameter, is of higher quality than "confounding" iterations involving changes to multiple parameters simultaneously. Safoutin (2003) also argues that rapid feedback of design performance following changes encourages an "iteration-rich" process, which he suggests is a positive behaviour.

Badke-Schaub and Gehrlicher (2003) collect a continuous protocol of discussions in a design department over 2 weeks, extracting situations in which the group worked together to make a decision. Identifying different iterative patterns, they found that a common characteristic of the productive situations—and a recommended design strategy—was clarification of goals before commencing. The development of more detailed mental models through the iteration process was also noted in the productive situations.

2.1.5 *Iteration may help to direct the design process*

Finally, iteration does not only involve development of the design, but also monitoring, self-reflection, and control of the design process. Adams and Atman (2000), for instance, find in a protocol study that cycles of monitoring progress occur alongside those of progressing the design. They conclude that the former are also important to navigate the design process effectively. In common with many protocol studies involving students, however, participants may need to learn how to approach the design process during the experiment itself, which is likely to affect the observed behaviour.

2.2 Insights from models of micro-level iteration

Complementing the empirical studies discussed above, micro-level iteration is presented as a defining characteristic of design in many abstract descriptive models. Such models are essentially domain-independent. Some have analysed design in terms of iteration amongst fundamental activity types such as analysis, synthesis, and evaluation (Asimow 1962). Others have presented the design and creative processes as a repeated cycle of divergence and convergence. According to these models, on each iteration several ideas are (or should be) generated before selecting the most promising (Howard et al. 2008). Yet other models present design as an iterative cycle between forms of logic, such as the production–deduction–induction (PDI) model of March (1984). Abstract models of design that highlight its iterative character are also often presented alongside empirical study of micro-level iteration. For instance, Schön and Wiggins (1992) draw on protocol studies to argue that architectural design may be viewed as a cycle of seeing–moving–seeing.

The above models each comprise small number of elements or steps. More detailed frameworks have also been developed to analyse design in terms of iterative transitions between activity and/or reasoning types. For example, the function–behaviour–structure (FBS) framework describes design as a process in which functions are transformed into structures, and comparison of the simulated performance of those structures to the desired performance leads to alterations (Gero 1990). Hybs and Gero (1992) describe the iterative repetition of these steps as a key element in an "evolutionary process" in which the design is progressively adapted as new information about it is generated. Maher and Poon (1996) model design exploration as a process in which problem and solution spaces coevolve through a genetic algorithm. On each iteration, the two spaces each influence the fitness function for the other. Jin and Benami (2010) develop a model of creative conceptual design based on a generate–stimulate–produce (GSP) cycle in which design entities are iteratively generated and integrated into the emerging design considering their FBS properties. They identify a detailed set of operations and processes for each of the GSP steps. For example, to generate, a designer might suggest, declare, suppose, etc. Hatchuel and Weil (2009) develop the CK theory of design, which presents it as an expansion process in which the designer traverses back and forth, i.e. iterates, between "knowledge space" and "concept space" using various logical operations to concurrently develop relevant knowledge alongside the "undecided" design concept(s) that are creatively generated. All these models suggest that iteration amongst the defined transitions may be disordered, because designers react to issues falling under their focus of attention and respond to new knowledge as it is gained. To summarise, many researchers are in agreement that iteration is central to design cognition, although exactly what is being iterated is described in different ways.

Models of micro-level iteration may also be found in contexts other than design research—any project by definition requires doing something new, once, and thus involves elements of problem-solving and iteration. This is recognised in the Deming/Shewhart PDCA (plan–do–check–act) cycle, which dates from the 1950s (Moen and Norman 2010). The PDCA cycle highlights that effective problem-solving is an iterative process in which adjustment of solutions based on feedback (check-act) is complementary to upfront analysis of the problem (plan–do). This cycle of feedback enabled through iteration is also emphasised in later problem-solving models. It may be especially important where the problems being solved are complex enough that they cannot be easily grasped, or in cases where the solution can influence the nature of the problem. This can occur, for instance, when creating products that define new markets or in design of long-life systems such as transportation, power, and health care—where there are many stakeholders having ambiguous wants and needs, and where the system's context of operation is likely to change over time. While micro-level problem-solving and design models both highlight beneficial effects of iteration, the former emphasise adjustment based on feedback while the latter focus mainly on how iteration enables creativity.

Overall, insights into micro-level iteration, gained through empirical studies and highlighted in models of the iteration process, highlight that it is necessary to account for information that emerges during the process—whether that information is generated by creative insights, from elaborating or learning more about the problem, or from interactions between the solution and its context.

2.3 Empirical insights into macro-level iteration

Insights have also been developed through empirical studies relating to iteration in the context of collaborative development. Most such studies focus on a single domain—either product development, construction management, or software project management. The insights are discussed in the next subsections. Similarities and differences across the domains are considered in Sect. 5.7.

2.3.1 An iterative project structure provides flexibility for a changing context

One of the main insights from research into micro-level iteration is that it is necessary to integrate information that emerges during the design process (Sects. 2.1, 2.2). This theme is also evident in empirical research into iteration on the macro-level, especially in software process management. For instance, Cusumano and Selby (1997) and Cusumano (1997) discuss how Microsoft uses a deliberately iterative process in which the product is integrated through a daily build and incrementally stabilised as a project moves forward. This strategy allows flexibility to adjust the emerging parts of the code, while also coordinating the initially difficult-to-define contributions from individual members of a large development team. They propose that this approach is particularly useful to allow innovative product development in a market that changes rapidly. Iansiti and MacCormack (1997) continue this theme with a case study at Netscape, explaining how a rapidly developing market can be addressed through an iterative development process in which companies sense the needs of customers, test solutions to those needs, and integrate the subsolutions, in a continuous cycle. They highlight the need to avoid committing to a solution too early, which implies iteratively overlapping the concept and detail phases. MacCormack et al. (2001) similarly argue that iteratively searching around the main sources of uncertainty to understand their impacts early in software projects helps to avoid costly rework later. They also argue for release of early versions of a product to accelerate the feedback process. These ideas also have support in the PD domain, for instance the survey reported by Terwiesch and Loch (1999) suggests that the negative impact of iterations on project duration is greater in cases where the specifications are frozen earlier in the project.

2.3.2 No universal relationship between iteration and project performance

Iteration has both positive and negative effects. However, empirical studies are inconclusive on whether increased levels of iteration accelerate or hinder the development process overall. Eisenhardt and Tabrizi (1995) draw on 72 projects in the computer industry to argue that increased iteration may accelerate a process, because (1) the chance of success is improved since each iteration offers more opportunity for a "hit"; (2) trying different variations on a design helps understand the impact of different parameters; (3) learning through iterations is quicker that what they term "more cognitive" learning strategies; (4) more consideration of alternatives reduces fixation; and (5) iterations improve the confidence of development teams, because they can feel less likely to have overlooked something important. On the other hand, providing evidence for the negative effects of iteration, Terwiesch and Loch (1999) survey 140 projects in the electronics industry. They find that number of iterations (where an iteration is defined as a modification of at least 10 % of the design) is positively correlated with a project's duration.

In software project management research, motivated by the popularity of agile methods with their prescriptions of iterative, incremental development, some authors have conducted empirical studies seeking to evaluate agile and its benefits against traditional waterfall methodologies that aim to reduce macro-level iteration by careful requirements definition, interface specification and stage gate reviews. Dybå and Dingsøyr (2008) review such studies, finding that some researchers conclude that more iterative agile methods lead to increased productivity and/or fewer defects—while others do not find evidence to support such claims.

The lack of a universal relationship between iteration and project performance is also evident in the construction industry. To illustrate, Love (2002) survey 161 building construction projects in Australia to elicit practitioners' judgements of rework costs, concluding that although 52 % of cost overruns are attributable to rework, schedule overruns could not be explained by amount of rework. Love et al. (2010) later survey another 115 projects, this time in civil construction. In this domain, they *did* find significant relationship between rework and schedule growth. Hwang et al. (2009) provide more evidence for variations in rework impact across different types of construction project, drawing on survey data of 359 projects.

Overall, studies in construction, software, and product development all imply that the impact of iteration on a project can be either positive or negative and is likely to depend on situation-specific factors.

2.3.3 Iteration is influenced by testing strategy

Empirical studies have indicated that companies are able to influence the iterative characteristics of their development processes by their choice of process structure. One element of this is the strategy for integrating testing into the development process. Thomke (1998) study this issue, undertaking a case study and survey of integrated circuit development. They focus on when companies move from virtual experimentation to more costly, but more accurate physical testing. When physical testing is very time-consuming and costly, designers try to work out as many problems as possible using simulation prior to creating a physical prototype. On the other hand, when physical testing is less costly, designers use it earlier in the design process. The survey revealed that this causes more iteration in early design but is also quicker and less costly overall.

2.3.4 Iteration is influenced by preliminary information

Much research on iteration has been done in the context of concurrent engineering, considering the iterative consequence of overlapping dependent activities. The basic idea is that overlapping phases of the development process (such as design and production) may reduce both lead time and the likelihood of lengthy, expensive iterations, for instance by involving production engineers early in the design process. However, increased concurrency can in itself cause more iteration because of the need for some activities to start work with information that is not yet finalised and that therefore may change later. Thus, some concurrent process designs may be better than others in terms of achieving an appropriate balance between these positive and negative effects.

Research on this topic has included a number of empirical studies. For example, in one early paper, Eastman (1980) analyse a case study at a defence contractor, identifying benefits of passing preliminary information to enable concurrency and some negative consequences as summarised above. Clark et al. (1987) draw on extensive studies in the auto industry to distinguish between two types of overlapping with different iterative consequences: batch model of overlapping, in which information is transferred in a few batches early and then updated leading to rework of the downstream task, and dialogue model of overlapping, in which the downstream task provides more frequent feedback on the early information through high-intensity iterative communication patterns, such that the task outputs are refined together in “a continual give and take”. The first approach focuses on information transfer while the second recognises the fundamentally iterative problem-solving cycle. Both situations involve iteration, but Clark et al. (1987) assert that the latter is more efficient. Terwiesch et al. (2002) analyse design decisions in the automotive industry, finding that iteration increases as the accuracy and/or stability of preliminary information decreases. Fernandes et al. (2014) study the determination of parameter values during jet engine preliminary design, identifying an iterative process in which preliminary values fluctuate within bands that become narrower as a solution is reached. They find that certain parameters change less frequently than others and exhibit greater average magnitudes of change. They also identify characteristic frequencies of adjustment for most parameters they examined.

2.3.5 Iteration is influenced by design characteristics

Iteration is not only influenced by management approaches but also by characteristics of the design under development. This is particularly apparent from empirical studies into engineering change, which has been defined by Jarratt et al. (2011) as “alteration made to parts, drawings or software once they have already been released” and could arguably be viewed as iteration over the modified parts. Engineering is often released and placed under configuration control substantially before the design phase of a complex program is complete, in this situation one important factor distinguishing post-release changes from iteration prior to release is that the process and impacts become documented and traceable. Like any design process, redesign is also likely to be iterative regardless of whether it occurs before or after design release.

One study that highlights the impact of design connectivity and margins on rework from an engineering change perspective is reported in Eckert et al. (2004), who conducted 22 interviews in a helicopter manufacturer. They find that changes (thus rework) propagate between components in a way influenced by tolerance margins of parameters, and also note that the cost of changes increases later in the design process as integration levels increase. Other authors have discussed how product characteristics that reduce sensitivity to change may help to mitigate rework impact. For instance, in an empirical study of integrated circuit design, Thomke (1997) finds that flexible technologies substantially reduce the cost of design changes, while in the software domain, MacCormack et al. (2001) also argue that attention paid to flexibility at the architecting stage yields reduced rework costs later in a project. Liker and Morgan (2006) find that rework is reduced in Toyota's case by an emphasis on design standardisation through shared architecture, modularity, and shared reusable components. Overall, they point out that reduced variability across designs leads to more predictable PD processes with less iteration. The benefits of modularity and standardisation for reducing iteration have also been identified in empirical studies in the construction sector (Thuesen and Hvam 2011). These characteristics may be more valuable for some parts of a design than for others, depending on factors such as the typical redesign effort for the part and its level of integration within the design (Sered and Reich 2006). Overall, appropriate modularity may decrease the iterations in platform projects by increasing the number of parts that can be reused instead of redesigned; it may also reduce life cycle cost by making a system easier to modify (Engel and Reich 2015). On the other hand, excessive modularity may increase interface complexity (Engel and Reich 2015), potentially leading to more iteration (see Sect. 2.4.4).

2.3.6 Iteration is influenced by network effects

The studies discussed above each focus on selected factors that impact iteration behaviours. In practice, many factors can come into play at once. Considering this, some researchers have sought to clarify the network of cause-and-effect relationships through empirical study. Amongst the most thorough work is reported in a series of articles that examine the causes and effects of rework in two building construction projects (Love et al. 1999; Love and Li 2000; Love and Edwards 2004). The researchers visited these projects several times per week throughout their durations, identified rework events and collected relevant data, and interviewed practitioners to understand rework events. In Love et al. (1999), insights on the chains of causes for individual rework events are combined to develop a causal loop diagram, indicating the mechanisms by which factors interact and stimulate or suppress rework. Their conclusions include that rework is caused through chains of events and that positive feedback mechanisms can exacerbate rework effects. Overall, these studies conclude that "an intricately 'complex' interwoven array of factors" contributes to rework occurrence (Love and Edwards 2004). To reduce rework impact, it is important to consider the whole system of interactions, not individual causes (Love et al. 1999). In the context of product development, several authors have similarly analysed causes and effects of iteration. For instance, Arundachawat et al. (2009) generate a list of rework causes by reviewing literature on predicting rework. Le (2012) and Browning (1998) develop networks of iteration cause and effect similar in concept to that of Love and Edwards (2004), although both considering some positive, as well as negative impacts of iteration. All these networks indicate likely cause-and-effect relationships, although the strength of each interaction is likely to be context-dependent and thus, as pointed out by Love et al. (1999), the networks should be viewed as qualitative starting points to decompose a particular situation, rather than as general-case causal models.

Braha and Bar-Yam (2004a, b) also examine the impact of network effects on iterative behaviours, in this case focusing on the network of information flows between tasks. Analysing four task networks from different development domains, they find that all four exhibit scale-free properties in which a few tasks have many information dependencies with other tasks, while the majority have only a small number of such links. They suggest that iterative characteristics are dominated by the central tasks, because any problems associated with them are likely to propagate and have disproportionate impact on the process overall. Braha and Bar-Yam (2004a, b) also show that tasks with many outgoing links tend to have only a few incoming links, and vice versa. Considering the set of tasks that mostly consume information, they find that tasks very rarely have a large number of incoming links. They attribute this to the increasing complexity of assimilating input information as the number of sources increases and conclude that "distributed PD networks limit conflicts by reducing the multiplicity of interactions that affect a single task, as reflected in the incoming links" (Braha and Bar-Yam 2007). Regarding iteration, they argue that "such architecture reduces the amount and range of potential revisions that occur in the dynamic PD process, and thus increases the likelihood of converging to a successful solution". On the other hand, "some tasks communicate their outcomes to many more tasks than others do and may play the role of coordinators" (Braha and Bar-Yam 2007). Regarding iteration, this "may improve the integration and consistency of the problem-solving process, thus reducing the number of potential conflicts" (Braha and Bar-Yam 2007). The negative correlation between indegree and outdegree leads to a structure in which tasks having high outdegree tend to connect to tasks having high indegree and vice versa; Braha et al. (2013) point out that this "disassortative" property implies a smaller number of cycles between tasks than would otherwise be the case. They conclude that indegree and outdegree characteristics arising from the difficulty of integrating information and the relative ease of broadcasting it may act to suppress iteration-inducing structures on the macro-level (Braha et al. 2013).

2.3.7 *Empirical studies of iteration at Toyota*

Finally, some researchers have described how practices observed at Toyota reduce the adverse impacts of iteration in product development. For instance, Ward et al. (1995) and Sobek et al. (1999) focus on set-based concurrent engineering (SBCE), in which decisions are delayed as far as possible and multiple candidate designs are taken forward concurrently until enough information is generated to eliminate infeasible alternatives. Design changes that expand the sets are avoided. Ward et al. (1995) describe this practice as "a deliberate effort to define, communicate about, and explore sets of possible solutions" within clearly defined constraints, and present the approach as fundamentally different to "point-based iteration". They argue that an important aspect of SBCE is to reduce the iterations associated with system integration, because the need for subsystems to be compatible is accounted for during the narrowing process. Toyota's approach as described by Ward et al. (1995) does require revisiting the multiple alternatives as they are taken forward, for instance developing multiple prototypes instead of just one. This is arguably another form of iteration—although Ward et al. (1995) point out it may be possible to do some tasks efficiently on many alternatives at once. Overall, the process can be viewed as front-loading iterations in the development process to reduce more costly rework later (Liker and Morgan 2006). Providing further insight into the applicability of these principles, Terwiesch et al. (2002) conclude from an automotive case study that a process depending on point-based iteration is appropriate if rework costs are low, if the information is subject to ambiguity, or if the negative impact of stopping to wait for information is high. Otherwise, SBCE may be more appropriate.

2.4 Insights from models of macro-level iteration

Macro-level iteration has also been considered in simulation models and mathematical models. This subsection focuses on insights and understanding extracted from, or demonstrated using, such models. Support for practice based on modelling is not emphasised because, as noted earlier, such work falls outside the scope of this article.

The models consider that essential properties of iteration are determined by the way each task or design problem is situated with respect to others. Key insights are discussed in the next subsections.

2.4.1 *Iteration is influenced by concurrency*

The relationship between iteration, preliminary information, and concurrency was considered in the empirical studies discussed earlier, and it has also been studied using mathematical models. In one early study of these issues, AitSahlia et al. (1995) use algebraic models of sequential task execution, fully parallel task execution, and partially parallel execution to study how concurrency increases the number of tasks that have to be redone if iteration occurs, resulting in a situation they describe as thrashing. The tipping point at which more concurrency ceases to be appropriate depends on the probability of coupled tasks creating iteration.

A number of authors develop mathematical models of how iteration is created when coupled tasks are forced to overlap. Krishnan et al. (1995) study how activities can be overlapped through preliminary transfer of information from an upstream task to allow a downstream task to be started early, and the iteration incurred on each subsequent update of that information. In their model, the duration of each iteration depends on the amount of change in the estimated point value, as well as the downstream task's sensitivity to change. Krishnan et al. (1995) show how the model can estimate the optimal amount of overlapping to minimise total time for the case of two tasks. Roemer et al. (2000) consider the trade-off in which the iteration caused by overlapping two tasks is balanced against the ability to do more work concurrently. They develop a model to find the shortest lead time for a given cost and vice versa. Joglekar et al. (2001) develop a model in which two coupled tasks each generate "design performance" at a constant rate while reducing the performance generated by the other, creating iteration to regain the prior level. They show how the relative rates of performance generation and the coupling strength between the tasks determine the optimal degree of overlap. They also show that if resource is shared amongst the two tasks, iteration impact increases relative to the situation where resource is available to execute both tasks at once.

The models discussed above are all algebraic in nature. Other authors study the relationship between iteration and preliminary information using Monte Carlo simulation, which is less constraining and allows study of larger problems. For instance, Bhuiyan et al. (2004) develop a model that, in addition to iteration because of preliminary information transferred between process phases, incorporates iteration needed to integrate engineering functions within those phases. Calibrating the model with a case study at an electronics manufacturer, they find that increased functional interaction can decrease cross-phase iteration caused by preliminary information and reduce overall effort and time, even though it requires more iteration within phases.

2.4.2 *Iteration is influenced by the testing strategy*

Another issue affecting the patterns of iteration in a project is the management and timing of testing. This has been examined through empirical study as discussed earlier, but also through mathematical modelling. Ha and Porteus (1995), for example, develop a model to study the optimal timing of design reviews in a concurrent process, considering that reviews have the benefit of uncovering errors before they incur downstream wasted effort (i.e. reducing iteration impact), at the cost of review set-up and execution time. They show that the optimal timing of reviews depends on whether the concurrency or quality issues dominate. Ahmadi and Wang (1999) extend the work, considering resource allocation alongside design reviews and showing how they can be scheduled to minimise the risk of missing technical targets due to iteration.

2.4.3 *Iteration is influenced by the freeze strategy*

Freezing parts of the design, or finalising parameters, is another facet of the design process that can influence iteration behaviours. Considering this issue, Krishnan et al. (1997b) develop a model in which a set of design parameters must be sequentially determined to minimise a set of performance parameters. In the model, each performance parameter is determined by a subset of the design parameters; minimising the first performance parameter fixes the corresponding set of design parameters and thus limits the options for minimisation of the second performance parameter. These constraints caused by sequential decision-making cause quality loss wherein the later task must accept a less-than-optimal decision to avoid revisiting the earlier one. Krishnan et al. (1997b) consider that iterations are performed to negotiate better solutions that improve overall quality, and identify properties of parameters in a design problem that allow them to be excluded from these iterations, thus reducing their dimensionality and accelerating them. The relationship between design freeze and iteration is also considered in the models of Keller et al. (2008) and Lee and Hong (2015), amongst others.

2.4.4 *Complexity and uncertainty increase iteration*

Browning (1998) writes that “tightly coupled, highly iterative processes can expect greater difficulty converging to an acceptable design under a given schedule and budget”. Complexity associated with interconnected design problems or tasks has accordingly been shown by a number of researchers to influence design time. For instance, Braha and Maimon (1998) consider the design process to be a series of stages that each revisit the design and create more information about it. They develop an entropic model to express complexity of the design within a stage according to its information content and show how it can be used to estimate design time for that stage (Braha and Maimon 1998, 2013). The model shows how increasing complexity increases the time taken on each occasion the design is revisited. The entropic model has been adopted and applied by a number of other authors (see Ameri et al. 2008 for a review). In another early article, Hoedemaker et al. (1999) develop a model that incorporates several iteration effects of how a design task is decomposed into modules. Their model implies that the more modules, the more incomplete information must be transferred, and thus the higher likelihood of integration problems and more iteration to correct them. Hoedemaker et al. (1999) conclude that a specific number of modules minimises the overall design completion time and, furthermore, that this number reduces as complexity of the design increases. Loch et al. (2003) simulate design processes comprising different numbers of tasks with random connectivity patterns, similarly finding that convergence takes longer with larger problem sizes until it becomes impossible. Yassine et al. (2003) show that increasing the coupling density in a design increases the amount of iteration in the design process and eventually causes instability; thus, reducing pairwise coupling leads to more rapid completion. Braha and Bar-Yam (2007) develop a model in which a PD process comprises a network of tasks that must all be solved. When a task is solved, there is a certain probability that this will cause each connected task to become unsolved and thus require iteration. Executing their model on the four PD task networks analysed by Braha and Bar-Yam (2004a, b), they use it to support and elaborate their empirical conclusions summarised in Sect. 2.3.6.

Other models examine the relationship between complexity and iteration using spectral analysis, building on the Work Transformation Model (WTM) developed by Smith and Eppinger (1997a). In their article, Smith and Eppinger (1997a) consider that at certain points in a development process, groups of strongly coupled tasks are executed in parallel with frequent information transfer to resolve dense cycles of information dependency. The model assumes that each task continuously creates iteration work for each of its coupled relatives according to the dependency strength between them. Smith and Eppinger (1997a) explain how eigenstructure analysis can be used to identify the drivers of iteration within a coupled task group if the WTM assumptions hold. Loch et al. (2003) also apply an eigenstructure analysis to study convergence of an iterative process, complementing their finding from simulation study described in the prior paragraph by showing that convergence becomes less probable and more time-consuming as the number of coupled tasks increases. Nonlinear effects in the convergence process mean that, as the critical complexity threshold is approached, small differences in the process can have a large impact on the iterative behaviour. This is highlighted in the work of Huberman and Wilkinson (2005) and Schlick et al. (2013) who both use extended WTM models to consider fluctuations in task performance. They apply results from linear algebra and autoregressive vector processes to show that convergence time of the iterative process becomes dramatically less predictable once uncertainty about task duration exceeds a certain level.

2.4.5 *Iteration is influenced by coordination*

Models discussed in prior subsections focus on information flows directly related to developing a design. Coordination research considers how this is interrelated with information flows needed to align and integrate the participants in a project (Suss and Thomson 2012). For example, Loch and Terwiesch (1998) focus on the communication that enables overlapping of two dependent tasks. They consider that holding meetings to communicate more frequently during the overlapping period reduces iteration impact, because each change released by the upstream task will require more work to be redone the later it is dealt with. Optimal policies concerning frequency of coordination meetings are derived algebraically, balancing the iteration avoided against the time required to hold the meetings. Yassine et al. (2003) consider the decomposition of interdependent design work across teams that share information only periodically and therefore have imperfect visibility of each others' progress. They show how this causes oscillatory iteration in which progress appears to be repeatedly on schedule prior to falling behind, arguing this causes several negative behaviours such as short termism in resource allocation. The model of Levitt et al. (1999) considers that when communication channels between departments become overloaded by too many coordination messages, designers need to make assumptions to proceed, which increases mistakes and the amount of iteration to correct them.

Mihm et al. (2003) and Loch et al. (2003) model design convergence as a series of iterations within which designers work concurrently to select values for design parameters. On each step, the model assumes that each designer chooses their parameter to minimise a local performance function that is also dependent on other connected parameters. Several coordination strategies are shown to accelerate convergence in this situation: ensuring each agent aims for the global performance function instead of the local one; accepting a lower level of design performance overall; reducing information transfer delays so that all decisions are based on up-to-date information on each iteration; and taking stepwise incremental improvements on each iteration, instead of having all agents attempt to jump simultaneously to their individually optimal solutions.

2.5 Summary

Many perspectives on iteration have been presented in the research literature, as reviewed in Sect. 2 and summarised in Table 1. Authors focusing on the micro-level have presented design and problem-solving processes as an iteration amongst fundamental types of activity. There is a consensus that micro-level iteration is essential in situations where the problem being solved is ambiguous or ill-defined, when many solutions might be possible, and when creativity plays a significant role. There is also evidence to support some cause-and-effect relationships, for instance between iteration over concepts and solution quality. However, empirical insights on these relationships are often developed from protocol studies in simplified situations, leading to questions of generalisability. Some of the studies are focused on insights for design education rather than practice and should be understood in this context. Another issue is that categorising process episodes into a small number of reasoning or activity types almost inevitably involves perceiving that process as highly iterative in nature. A potential topic for future research is therefore to examine in more detail the degree to which iteration is an artefact of the process description or experimental method, versus a characteristic of the process itself.

On the macro-level, a number of observational, rather than experimental, empirical studies have been reported in the domains of product development, software, and construction management, yielding a consensus on several points as summarised in Sect. 2.3. However, there is disagreement on other points as indicated for example in Sect. 2.3.2. Numerous task- and dependency-oriented models have also been developed to cast light on iteration. Strong simplifying assumptions are often made in these models, in many cases with limited empirical support. Quite often, the main justification is mathematical tractability. Considering this issue, Smith and Tjandra (1998) compare the patterns of iterations shown in their empirical study (discussed earlier) with some of the models discussed in this subsection, finding significant divergences from the work of Smith and Eppinger (1997a, b), AitSahlia et al. (1995), and Krishnan et al. (1997a) in how the time required for iterations is determined, the patterns of concurrency, and the stochastic nature of iteration initiation. It should be noted, though, that because the focus of the mathematical models is primarily on developing formalisms and theoretical insights, a critique of applicability to practice might not always be appropriate.

3 Iterative stereotypes

The literature review in Sect. 2 highlights that researchers have not only developed different insights about iteration, but also have described iteration processes in quite different ways. This is because different situations may be dominated by different iterative characteristics. At the same time, iteration may be legitimately viewed from different perspectives depending on the concerns at hand. Nevertheless, the multiple perspectives and lack of clarity regarding their relationships can make it difficult to determine how to handle iterative situations in practice.

The remainder of this article draws upon the review to introduce the concept of *iterative stereotypes* to summarise and relate these different descriptions of iteration processes. An iterative stereotype is a simplified depiction of iteration which can concisely express selected characteristics of an iterative situation. A taxonomy of stereotypes is expected to clarify the possible ways to describe iteration and thus help to recognise important characteristics of processes in practice.

3.1 Iterative stereotypes in the literature

Insight into iterative stereotypes can be gained from the terminology authors use for describing different iteration processes. We extracted twenty-one distinct sets of terminology from our literature review, which are collected in Table 5. Some of the key points are summarised below.

First, a quite common view is that iteration can or should be distinguished according to its good or bad effects. For example, Clausing (1994) discusses how iteration can be either *Creative* or *Dysfunctional*. Browning (1998) discusses two types of iteration: *Intentional*, which he equates with enabling convergence or learning, and *Unintentional*, resulting from “new information arriving at the wrong time”. Yassine and Braha (2003) similarly distinguish *Planned iteration*, which occurs when assumptions are knowingly made to progress a design as part of a convergence process, from *Unplanned iteration* due to “unexpected failure of a design to meet specifications”. Le (2012) describes these cases as *Progressive* and *Corrective,* respectively. Haller et al. (2014) describe iteration as either *Creative* or *Superfluous*, where the former type accounts for new customer requirements and is said to add “quality”. Haller et al. (2014)’s definitions differ from others in this paragraph in emphasising that the desirable versus undesirable distinction is not inherent to the iteration, but to the responsible party.

Arguably, each of these authors makes a similar underlying distinction between positive and negative effects of iteration, although the terminologies differ. However, several studies reviewed in Sect. 2 indicated that iteration can incur positive and negative effects simultaneously. In practice, it might therefore be difficult to delineate some iterative situations in this way (Le 2012).

Second, some authors emphasise a difference in scale, which inspired the micro- versus macro-level distinction used in Sect. 2. For example, Eppinger et al. (1994) discuss three iterative situations delineated by the increasing scope of feedback loops that cause them: (1) *Concurrent iteration* to develop tightly coupled subsystems through frequent information exchange; (2) *Rework*, which they view as costly long iterations caused by system-level feedbacks; and (3) *Generational learning*, where feedbacks occur too late to influence the current project. Safoutin and Smith (1996) differentiate between (1) *Micro-scale iteration* which they describe as computational iteration to minimise an error term, (2) *Meso-scale iteration* which they describe as “the proposal, testing and modification cycle”, and (3) *Macro-scale iteration* which they view as repeating an entire design process to generate a new version of a product. Chusilp and Jin (2006) also focus on differences in scale in delineating (1) *Design task iteration*, which indicates revisiting tasks in a project, often involving many members of a design team; from (2) *Mental iteration*, which focuses on the cognitive cycles involved when an individual designer addresses a design problem.

A third group of terminologies emphasise the type of changes in the design or the tasks being performed. For example, Isaksson et al. (2000) propose that iterative situations may be positioned on a 2 × 2 grid. On the first dimension, *Repetitive iteration* describes situations in which a revisited activity remains the same, whereas in *Evolutionary iteration* the set-up of the activity changes. The second dimension of their grid emphasises that in both cases, the iteration may be *Intentional* or *Unintentional*. Costa and Sobek (2003) consider: (1) *Rework iteration*, in which an activity is revisited at the same abstraction level and on the same part of the design to correct errors made earlier; (2) *Design iteration*, in which part of the design is revisited at an increasingly detailed level of abstraction; and (3) *Behavioural iteration* in which the same task and abstraction level are revisited, but the scope changes, e.g. because the designer considers a different product subsystem.

Fourth and finally, some terminologies do not fit into any of the other groups discussed above. For instance, focusing only on the micro-level, Adams and Atman (2000) consider two situations: *Diagnostic iteration* involves cycles of monitoring performance and learning, while *Transformative iteration* involves integrating new information and altering the design. Wynn and Eckert (2007) provide six terms to describe iterative situations, summarised in Table 5. They describe these terms as non-orthogonal, arguing that iteration can be perceived in different ways. They give the example of a manager and a designer who disagree over whether iteration adds value. Overall, the terminologies we identified are mainly introduced to make particular points in the respective publications, usually not supported by explicit research methodology. Exceptions include the work of Le (2012) who develops his terminology from literature review, and Browning (1998) who develops his analysis by integrating literature review with empirical study. We did not identify any terminology that considers all the situations identified in our review—although certain key ideas do appear repeatedly, as outlined in this subsection and in Table 5.

3.2 Developing an integrating taxonomy

A new and more comprehensive taxonomy was developed to address these shortcomings. This was done by integrating the ideas from the literature reviewed in previous sections. The terminology introduced by Wynn and Eckert (2007) was selected as the starting point because it initially appeared the most comprehensive. The other sets of terminology summarised in Table 5 were then considered to determine whether their concepts were covered by Wynn and Eckert (2007). This yielded substantial improvements including the expansion from six to 13 stereotypes, all of which are discussed in prior research, although they have not been previously brought together and articulated as a set. After several cycles of adjustment, the new taxonomy was able to account for almost all iterative situations revealed by the literature study. Exceptions are discussed in Sect. 5.8.

We also sought to arrange the 13 stereotypes into groups to assist comprehension. To do this, we focused on the distinction between primarily positive and primarily negative iteration, on the basis that this is already well established in the literature (see Sect. 3.1). A third group was introduced to encompass stereotypes that emphasise positive and negative effects in combination. This provided a useful organising framework for Sect. 4, although other groupings of the stereotypes would be possible.

4 A taxonomy of iterative stereotypes

The taxonomy organises iterative stereotypes according to function, conveying what is achieved or intended by revisiting tasks, problems, or aspects of a design. Three distinct iterative functions were identified: (1) to *Progress* towards completion; (2) to *Correct* errors introduced earlier or to implement a change; or 3) to help *Coordinate* actors, decisions, and/or flows of work. These functions and the stereotypes that embody them are introduced in the next subsections with reference to key publications. Section 4.4 then considers how the entire taxonomy maps onto the literature reviewed in Sects. 2 and 3.

4.1 Progressive iteration

Design and other problem-solving activity creates information and knowledge (Hatchuel and Weil 2009). As understanding is generated, issues must often be revisited to consider them in the light of insights that did not previously exist (Yassine et al. 2003). This is an example of progressive iteration, which many authors agree cannot be avoided entirely during design. Factors contributing to progressive iteration include (1) uncertainty in the problem and/or solution definition, including incompleteness or ambiguity at the time decisions are made; and (2) the bounded rationality of problem solvers, meaning they cannot reason about a complex interconnected system of issues simultaneously, and therefore must decompose it into parts that are addressed one at a time. In the taxonomy, the progression stereotypes suggest iteration is necessary because of these two conditions.

Cyclic dependencies between parts of a problem are also sometimes presented as a cause of progressive iteration. For example, Evans (1959) gives the example of bridge design in which the structure depends on the load, but the load depends on the static weight of the structure. Evans (1959) argues that this requires iteration to converge on a solution. More generally, there are several ways to resolve a cyclic problem structure, and not all of them involve iteration. Options may include (1) converge iteratively by passing parameters at a single level of detail; (2) converge iteratively through increasing levels of definition or detail; (3) break the interdependencies that cause the cyclic structures by selecting suitable value(s) for some of the interdependent parameters—which may not entail iteration if those values are carefully chosen; or (4) solve the problems as a set through appropriate mathematical manipulations, where possible.

Five stereotypes of progressive iteration were identified in the literature. They are depicted in Fig. 2 and discussed below.

4.1.1 Exploration

Exploration refers to the concurrent and iterative initial development of problem and solution, which is generally seen as fundamental to the creative problem-solving process (Sect. 2.2). The exploration stereotype emphasises the initially ill-defined nature of goals and activities (Simon 1973), leading to a disordered process as designers iteratively discover, structure, and address emerging issues. Exploration as defined here occurs at a fixed level of design definition. For example, it may describe the iterative evolution of a concept represented in a series of sketches.

4.1.2 Concretisation

The concretisation stereotype refers to an iterative process in which constituent elements of a design are firmed up by revisiting them at increasing levels of definition, while ensuring consistency amongst those levels (Safoutin 2003; Costa and Sobek 2003). For example, parts in a mechanical assembly may first be defined as simple solid models, which are later revisited to add details. Concretisation is concerned with progressively reducing ambiguity by creating more information about the design (Braha and Maimon 1998). It may involve concurrent elaboration of the problem specification and may happen in disordered, opportunistic sequence (Guindon 1990).

4.1.3 Convergence

The convergence stereotype focuses on iterating to determine suitable parameter values and/or to adjust details such that well-defined performance objectives can be met. It emphasises point-by-point adjustment towards a satisficing solution, essentially in a monotonic fashion. This can be viewed as progressive increase in the precision of parameter values; progressive reduction in the number of problems remaining to be solved (Yassine et al. 2003); or iterative adjustment to take into account additional test results or constraints such as operating conditions, such that the required changes become progressively more minor.

Convergence emphasises optimising details once the main form of a design has been determined at a certain level of definition. It is often associated with complex relationships between tasks, parameters, and/or objectives. More sophisticated methods and/or tools may be applied as increasing levels of confidence are reached during convergence (Evans 1959), although as defined here, in comparison with concretisation the parameters or details being adjusted remain essentially the same.

4.1.4 Refinement

The refinement stereotype describes situations in which solutions that appear to meet primary objectives undergo further iterations to enhance secondary characteristics, for example to improve elegance or to reduce cost. In the engineering context, Wynn and Eckert (2007) suggest that excessive refinement may occur in cases where it is not obvious when to stop working on a problem, for example if there are few milestones to anchor a development schedule; if additional time is available; or if evaluation criteria are subjective. In software, refactoring is a form of refinement that involves incrementally modifying the internal structure of a system without changing its external behaviour, in order to improve attributes such as maintainability, extensibility, and simplicity (Mens and Tourwé 2004; Kim et al. 2014).

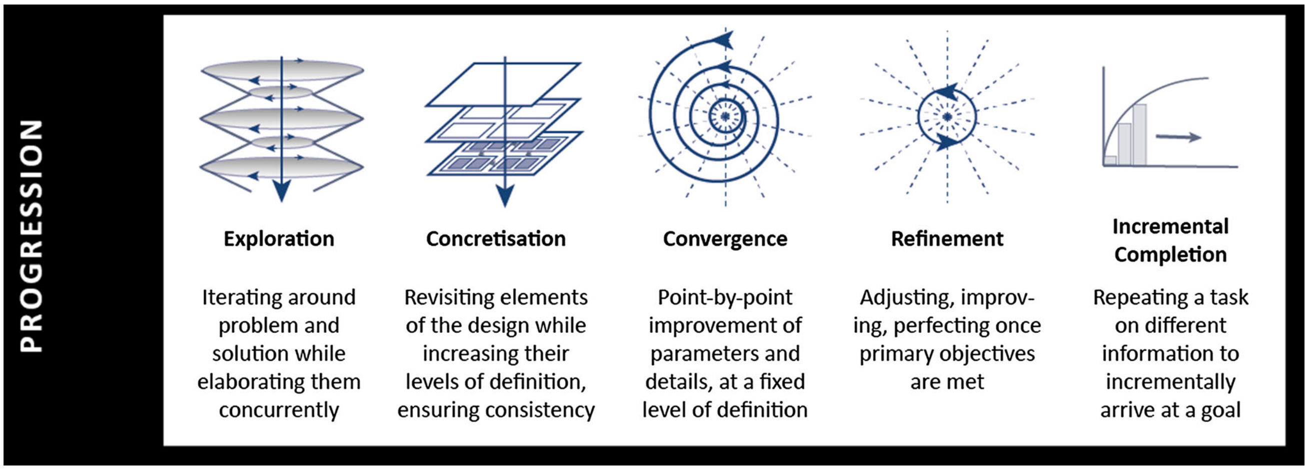
PROGRESSION
Exploration
Iterating around problem and solution while elaborating them concurrently
Concretisation
Revisiting elements of the design while increasing their levels of definition, ensuring consistency
Convergence
Point-by-point improvement of parameters and details, at a fixed level of definition
Refinement
Adjusting, improving, perfecting once primary objectives are met
Incremental Completion
Repeating a task on different information to incrementally arrive at a goal

Fig. 2 Stereotypes of progressive iteration

4.1.5 Incremental completion

Incremental completion refers to the planned repetition of a task or process to gradually move towards a desired goal. For example, Safoutin (2003) introduces this stereotype using the example of a numerical integration loop. Costa and Sobek (2003) explain it in terms of decomposing a design into subsystems which may each be developed by different teams that follow similar processes, prior to integration. Some forms of iterative incremental development divide a software system into segments of functionality, which are incrementally implemented and integrated through a planned series of iteration cycles (Larman and Basili 2003). This is also a form of incremental completion.

4.1.6 Comparison of progression stereotypes

Table 2 summarises key differences between the five progression stereotypes. Considering the first four stereotypes, whether a design (or part of a design) is progressed through exploration, concretisation, convergence, or refinement depends in part on its maturity. In exploration, decisions are loosely constrained and iterations may lead to qualitatively different solutions. In concretisation, progress is more constrained by previous decisions. When convergence or refinement is undertaken, main solution parameters are already delineated at the level of definition on which the iteration occurs. Iterations are undertaken to determine suitable values for them and/or to adjust details. Whereas exploration and concretisation involve substantially qualitative and subjective reasoning, convergence and refinement typically emphasise quantitative analysis and may focus on parameter values and constraints.

These differences are reflected in the structure of the iteration process. Stereotypes depicted towards the left of Fig. 2 emphasise an initially responsive and unstructured process, while at the other end of the spectrum, stereotypes towards the right of the figure indicate a repeated sequence of similar operations. In all cases, the detail of tasks may differ from one iteration to the next according to the evolving situation.

Incremental completion differs from the other progressive stereotypes discussed above in that a task or process is repeated on different aspects of the design, rather than revisiting the same aspects of the design using different tasks, on different levels of detail, or considering new relevant information.

4.2 Corrective iteration

The second function, correction, describes iteration in response to unplanned adverse events. It is associated with new information that reveals problems in the design (Sobek et al. 1999). As such, it is often associated with system integration activities, testing, and engineering changes after design release. The latter might, for example, be initiated to account for requirement changes or to respond to issues arising later in the life cycle, for instance if a part needs to be redesigned following failures in service (Ahmad et al. 2013). From another point of view, tasks may be revisited to check and perhaps regenerate their outputs after input information is updated (Wynn et al. 2014).

Corrective iteration is generally undesirable because it would not be required if those adverse events had not manifested. However, correction can still provide positive effects, for example if additional knowledge is generated or the changes that are made add value to the design (Haller et al. 2014).

The three stereotypes of corrective iteration that were identified are summarised in Fig. 3 and described in the next subsections.

4.2.1 New work

The new work stereotype, appearing in Taylor and Ford (2006) and Isaksson et al. (2000), focuses on correction in which a solution is revisited to meet requirements in a different way, i.e. through different solution principles that require a different work approach. It may involve initiating work to address issues that were not initially recognised. An example of new work is the replacement of a centrifugal compressor with an axial compressor, whose

Table 2 Comparison of progression stereotypes

	Level of design definition during iterations	Sophistication of tools/methods used during iterations	Solution viable during iterations?
Exploration	Remains similar	Same	No
Concretisation	Increases	May increase	No
Convergence	Remains similar	May increase	No
Refinement	Remains similar	May increase	Yes
Incr. completion	Increases	Same	No

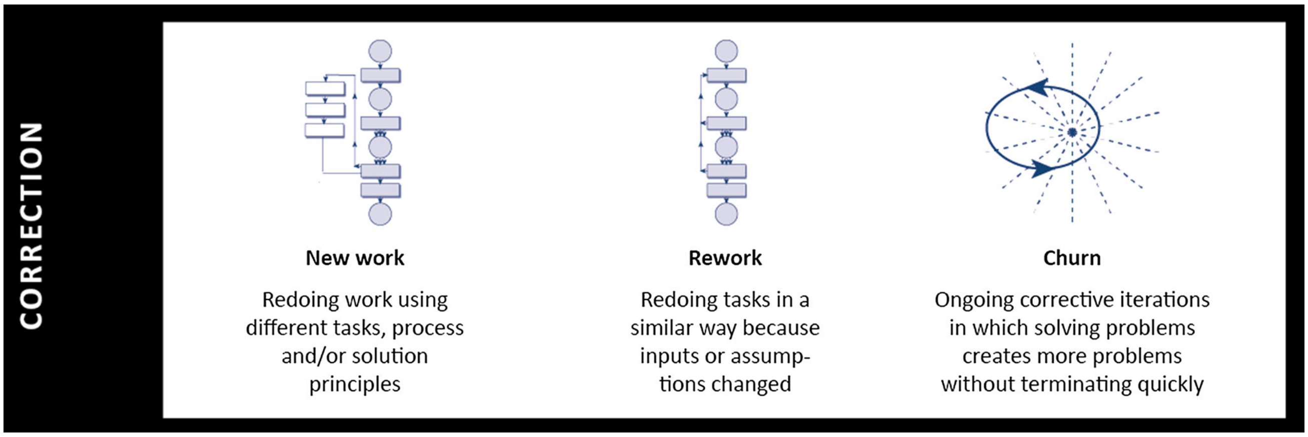
CORRECTION
New work
Redoing work using different tasks, process and/or solution principles
Rework
Redoing tasks in a similar way because inputs or assump-tions changed
Churn
Ongoing corrective iterations in which solving problems creates more problems without terminating quickly

Fig. 3 Stereotypes of corrective iteration

integration may require consideration of some different design issues.

4.2.2 Rework

Rework is one of the stereotypes that appears most frequently in the literature. It is described by Smith and Eppinger (1997a) as "the required repetition of a task because it was originally attempted with imperfect information (assumptions)". In comparison with new work, the approach used is similar to the last time the work was performed—although this distinction is necessarily subjective because it depends on the level of resolution to which the process is considered. An example of rework in product development is the modification of a drawing to correct a drafting error. As well as arising following new information about problems in the design, rework may also be caused if a process is too complex to identify the most efficient order of work execution (Eppinger et al. 1994), or because "the wrong information arriv[es] at the wrong time" (Browning 1998). Despite appearing frequently in the literature on iteration, acknowledgement and measurement of rework has been said to be the biggest gap in conventional project management techniques (Cooper et al. 2002).

4.2.3 Churn

The churn stereotype focuses on situations in which parts of a problem are revisited repeatedly without converging, because the parts interact such that each attempted solution creates another set of problems. It appears, for example, in the work of Mihm et al. (2003), Loch et al. (2003), and Yassine et al. (2003), which is discussed in Sect. 2.4.4. Churn can be viewed as failed convergence in the terminology of Yassine et al. (2003), who define the latter as a process in which the number of problems being solved reduces towards "an acceptable threshold". Alternatively, it can be seen as self-propagating rework. Churn is often associated with design complexity. Other factors contributing to churn may include decomposition of work across teams who share information only periodically, exogenous changes, imperfect testing, and oscillatory allocation of resources due to firefighting (Yassine et al. 2003).

4.3 Coordinative iteration

Coordination, which is the third and final function in the taxonomy, describes iteration associated with structures and approaches intended to make a process more effective, efficient, and/or predictable. Thus, for example, coordinative stereotypes convey iteration associated with set-based design, forced overlapping of dependent tasks, and agile development approaches. Coordination stereotypes imply that iteration is expected to provide benefits, such as reducing the amount or risk of more costly iteration elsewhere, that outweigh its cost, effort, and/or time implications.

Five stereotypes of coordinative iteration were identified from the literature study. They are summarised in Fig. 4 and Table 3, and described below.

4.3.1 Governance

The governance stereotype refers to iterations deliberately introduced to allow frequent information release from a process, facilitating its oversight and management. Unger and Eppinger (2011), for example, describe how iteration is used as a risk management process of "controlled, feedback-based redesign" governed by design reviews. Ahmadi and Wang (1999) consider the introduction of design reviews to regulate risk, which can be viewed as repeating review tasks as well as the tasks to prepare required

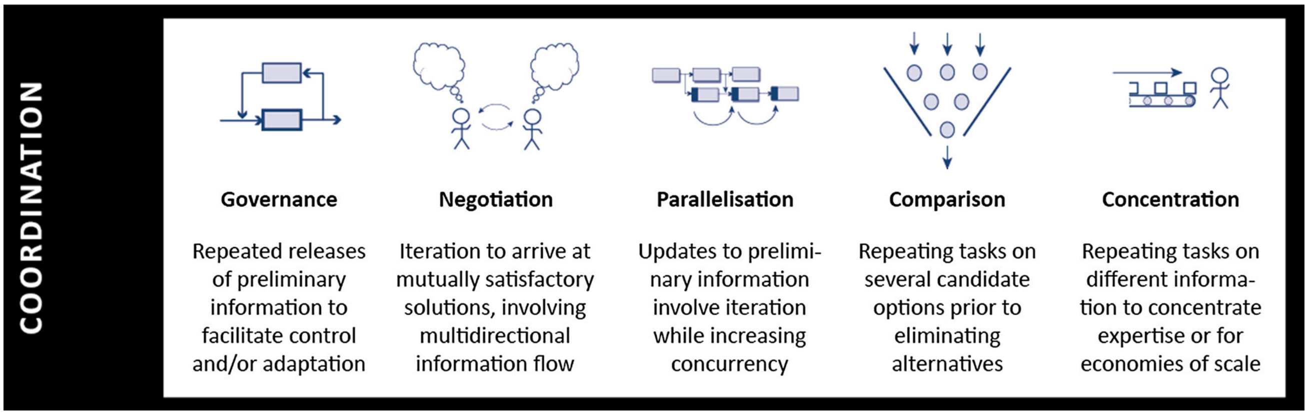
COORDINATION
Governance
Repeated releases of preliminary information to facilitate control and/or adaptation
Negotiation
Iteration to arrive at mutually satisfactory solutions, involving multidirectional information flow
Parallelisation
Updates to preliminary information involve iteration while increasing concurrency
Comparison
Repeating tasks on several candidate options prior to eliminating alternatives
Concentration
Repeating tasks on different information to concentrate expertise or for economies of scale

Fig. 4 Stereotypes of coordinative iteration

Table 3 Comparison of coordination stereotypes

	Typical motivation(s) for introducing the iterative structure	What is revisited?	Info. maturity	Typical destination of info.	
Governance	Control and/or adaptation	(Part of) a design	Increases	Management	Same phase
Negotiation	Facilitate integration	(Part of) a design	Increases	Other teams	Same phase
Parallelisation	Increase concurrency	(Part of) a design	Increases	Other teams	Next phase
Comparison	Improve quality/reduce risk	Task or process	Stays similar	Same team	Same phase
Concentration	Reduce costs	Task or process	Stays similar	Other teams	Next task

information. Structuring a project to provide frequent releases for feedback from an evolving context or from users (e.g. Iansiti and MacCormack 1997) may also be considered governance iteration. This stereotype also describes iteration structures enabling feedback to ensure completion on time, budget, and quality. For instance, information release to financial approval processes regulates expenditures, while iteration involving project management processes regulates design progress and schedule risks.

4.3.2 *Negotiation*

The negotiation stereotype describes situations in which an iterative process is used to allow trade-offs between goals and constraints owned by different participants to be understood and mutually acceptable solutions to be found (Bucciarelli 1994). This also appears in the dialogue model of overlapping discussed by Clark et al. (1987). Negotiation is characterised by multidirectional information transfer amongst the people involved.

To give some examples, an iterative negotiation process is used to integrate contributions from personnel who have a limited understanding of each others work, because of disciplinary specialisation, because individuals' responsibilities focus on certain parts of a design or certain aspects of a problem, and/or because a design is developed in collaboration across several teams or companies. It can involve conflicting objectives and can also appear where people do not know what they can achieve at the start of the design process (Eckert et al. 2014). Negotiation may also be necessary when multiple groups share responsibility for the same design parameters. In such situations, setting these parameters to maximise a local criterion for one group may deteriorate the solution for others (Mihm et al. 2003). In common with many other forms of iteration, negotiation becomes more difficult as the number of parameters involved increases (Krishnan et al. 1997b).

4.3.3 *Parallelisation*

The parallelisation stereotype refers to iteration associated with overlapping of consecutive phases or tasks in the development process. This involves repeating parts of the upstream work to communicate updates as the upstream task converges during the overlapping period, as well as revisiting elements of the downstream work to account for the successive changes. This stereotype features in several of the models reviewed in Sect. 2.4.1, e.g. Krishnan et al. (1995), as well as empirical studies such as that reported by Terwiesch et al. (2002).

4.3.4 *Comparison*

In many situations during a project, several alternatives are considered in parallel until there is enough information to decide between them. In this case, some activities may need to be repeated for each candidate taken forward, which is a form of iteration. This stereotype appears in the literature on SBCE discussed in Sect. 2.3.7. As noted by Sobek et al. (1999), who use the term "narrowing iterations", it is also evident in textbook models that portray early stages of the development process as a funnel, such as Ulrich and Eppinger (2011). Comparison also appears in work which emphasises iterative generation, comparison, and recombination of multiple concepts during early design (e.g. Chusilp and Jin 2006).

4.3.5 *Concentration*

Concentration refers to iterations that occur after organising a process so that a task or process is repeatedly executed on different information (Wynn and Eckert 2007). This stereotype may be appropriate, for example, to describe situations where jobs of a similar type are organised to flow through a single department, or collected into batches and done together, to reduce set-up and coordination overheads. In such cases, similar tasks that were previously done on different parts of the design by different people are brought together such that a single group repetitively performs, i.e. iterates, the task. In this situation, the task remains similar but is done on different information each time. Concentration iteration may help obtain economies of scale, may be part of outsourcing, or may allow sharing of limited expertise in a particular area across projects that need it. The implications of concentration on process performance can be significant and are discussed in detail in the literature on lean product development (e.g. Oppenheim 2004; Beauregard et al. 2008).

4.4 Verifying the taxonomy

To recap, the motivation for developing the taxonomy was to (1) incorporate and (2) distinguish between the iterative situations revealed in the literature study. To verify that these objectives were met, the literature from this article's bibliography was revisited to ensure that the discussions in each publication could be appropriately aligned against one or more stereotypes. The result of this mapping process is shown in Table 6.

Compiling Table 6 required detailed and systematic attention. We proceeded as follows. First, because of the profusion of terminology, we focused on authors' descriptions of iteration rather than the specific terms they use. Furthermore, in the case of models, we focused on the authors' descriptions of the situation rather than the detail of its representation. Although the taxonomy integrates key concepts found in the literature, we found that authors' descriptions of iterative situations are sometimes ambiguous or lacking contextual description, and in some cases were initially difficult to align against a single stereotype. To address this issue, decision guidelines were formulated from the stereotype definitions to crystallise key differences between them. The guidelines are shown in Table 4. They helped to ensure consistency when assigning each described situation to a single stereotype, although the need for interpretation could not be eliminated entirely.[1]

Certain publications were deemed to describe multiple distinct situations. Such cases are mapped to multiple stereotypes in Table 6. Three publications describe iterative situations that relate to generational learning and are thus outside the scope of the review as defined in Sect. 1, although other elements of those publications are in scope. These cases are discussed in Sect. 5.8.

Finally, after completing the mapping, the stereotype descriptions were reconsidered to improve their clarity. To do this, we worked down the columns of Table 6 to identify papers that had been mapped to each stereotype and then considered those publications again to verify that the stereotype descriptions and any relevant decision guidelines were appropriately formulated. This yielded some final adjustments.

Overall, Table 6 aligns each stereotype against publications that discuss it and thereby provides an organised source of reference to support the taxonomy. Table 6 also provides a starting point to navigate and analyse the literature. For instance, reading down the concretisation column of the table indicates key literature relevant to that stereotype and shows the development domain considered by each of those publications.

Expanding upon the final section of Tables 6, 7 positions the terminologies collected in Table 5 against the stereotypes. This highlights the lack of agreement on terms and confirms that prior publications which propose terminology only cover subsets of the iterative situations that are collected in this article.

5 Discussion

Complementing the summary of issues and the integrating taxonomy, a number of key insights, implications, and opportunities for future research were drawn from the review. These are summarised in the next subsections.

[1] For example, some articles include extremely brief mention of particular iterative situations. We used our judgement to determine whether such cases should be mapped onto the corresponding stereotypes in Table 6.

Table 4 Clarification of the similarities and differences between selected stereotypes

Decision	Similarities between the stereotypes and guidelines to decide between them
(Exploration or concretisation) versus (convergence or refinement)?	*All these stereotypes emphasise developing knowledge and progressing a design.* Guideline: exploration and concretisation both emphasise qualitative reasoning and elaboration of objectives during the iteration process. Convergence and refinement both emphasise adjustment of defined parameters or details and may involve more quantitative reasoning
Exploration versus concretisation?	*Both are associated with qualitative reasoning and/or ambiguity in early design.* Guideline: If an increase through levels of design definition is emphasised, assign concretisation. If iterations take place at the same level of definition (e.g. a concept sketch), assign exploration
Parallelisation versus negotiation?	*Both relate to iteration associated with concurrency.* Guideline: If the iteration is associated with updated preliminary information that flows unidirectionally from an upstream to a downstream activity, assign parallelisation. Otherwise, if multidirectional flow amongst two or more parties is emphasised, assign negotiation
Convergence versus negotiation?	*Both may relate to iteration required to resolve complex interdependency.* Guideline: If the need for iteration because different aspects of an interdependent design problem are owned by different people is emphasised, assign negotiation. If iterating to resolve interdependency amongst tasks, parameters, or decisions is the main focus and iterating to communicate is not emphasised (even though it may be a team activity), assign convergence
Incr. completion versus concentration?	*Both concern repetition of a task on different parts of the same design.* Guideline: If the purpose of repeating the task is emphasised to be completing the design, assign Incr. completion. If the reason is mainly that a person or department is specialised in that work, assign concentration

5.1 Iteration entails both good and bad effects

Some iteration adds value by facilitating progressive generation of knowledge. Some may be unavoidable. Even apparently unproductive iteration can have beneficial side effects when viewed from another perspective. For instance, increased iteration in a development process can reduce its overall duration if it allows more work to be attempted concurrently, or if it allows interdependent tasks to be overlapped. This is sometimes recognised by describing iteration as either essentially desirable or essentially undesirable (Sect. 3.1), while other research highlights that iteration can have these effects simultaneously.

Desirable and undesirable effects of iteration are reflected across the different functions in the taxonomy. Progressive iterations directly create knowledge and value (although they also incur additional time, effort, and/or cost). Corrective iterations only occur because of issues that would be better avoided at their source. Coordination iteration adds value by enabling secondary and higher-order effects, such as enhanced control of schedule, cost, and risk. The undesirable impacts of iteration are usually apparent, although they may be difficult to quantify. However, the benefit iteration may provide can be easily obscured by the complexity and ambiguity of interactions, especially in large projects.

5.2 Iteration entails uncertainty and is self-reinforcing

Iteration effects are amplified by uncertainty in products (Liker and Morgan 2006) and in processes (Huberman and Wilkinson 2005). A more iterative project may be less efficient, because as uncertainty increases, so does the number and severity of exceptions from standard process that are generated, and so does the communication and management overhead needed to handle them (Galbraith 1974). Thus, a project with more iteration typically involves more ad hoc coordination and consequently more opportunities to make mistakes or overlook important issues—requiring further iteration to address. In other words, iteration can reinforce itself through positive feedback (Love et al. 1999). For this reason, reducing unnecessary iteration in a project may have knock-on benefits beyond the elimination of repeated work.

5.3 Iteration may be perceived in different ways

The variety of perspectives discussed in Sects. 2, 3, and 4 highlights that there are many possibilities for perceiving iteration. One consideration is that the perceived amount and form of iteration can be influenced by an individual's viewpoint, responsibilities, and objective (Wynn and Eckert 2007). For instance, designers and managers may disagree on whether a particular iterative situation is progressive and necessary, or corrective and perhaps avoidable. This subjectivity might make it difficult to find an appropriate way to respond to and/or plan for iterative situations. For instance, it might be appropriate to plan for convergence, e.g. by leaving buffers in the schedule or explicitly planning several cycles, while rework might be better addressed through identification and mitigation of likely causes. However, such actions might be difficult to

Table 5 Terms used to differentiate between iterative situations

References	Description of terminology
Various authors	Terms used to describe iteration having essentially desirable versus essentially undesirable effects by Clausing (1994)/Browning (1998)/Ballard (2000)/Yassine et al. (2003)/Le (2012)/Haller et al. (2014) are, respectively:
	Creative/intentional/positive/planned/progressive/creative iteration
	Dysfunctional/unintentional/negative/unplanned/corrective/superfluous iteration
Eppinger et al. (1994)	*Concurrent iteration*—concurrent development of tightly coupled subsystems, requiring substantial coordination
	Rework—costly, long iterations caused by system-level feedback
	Generational learning—Iterating product versions when feedback is too late to influence a project
Safoutin and Smith (1996)	*Micro-scale iteration*—computational iterations to minimise an error term
	Meso-scale iteration —"The proposal, testing and modification cycle"
	Macro-scale iteration—repeating an entire design process to generate a new version of a product
Adams and Atman (2000)	*Diagnostic iteration*—cycles of monitoring performance and learning
	Transformative iteration—cycles of integrating new information and altering the design
Isaksson et al. (2000)	*Intentional repetitive iteration*—planned to converge or compare options; criteria and task stay the same
	Unintentional repetitive iteration—needed because specifications not met; criteria and task stay the same
	Intentional evolutionary iteration—planned to converge or compare options; new information alters task set-up
	Unintentional evolutionary iteration—needed because specifications not met; new information alters task set-up
Costa and Sobek (2003)	*Design iteration*—an activity is repeated on the same part of the design but at a different level of abstraction, e.g. detail design instead of concept design
	Behavioural iteration—a task is applied at the same abstraction level to a different part of the design. For instance, stress analysis is repeated on several components in an assembly
	Rework—to correct errors, an activity is revisited on the same scope and abstraction
Safoutin (2003)	*Repetition*—moving towards a goal incrementally by doing the same thing multiple times, e.g. the loop of numerical integration to calculate the area under a curve
	Progression—refining or gradually arriving at a solution that is whole on each attempt but in some sense "better" each time. Each successive solution represents an intermediate approximation of the goal
	Feedback—progression that is guided by feedback regarding the suitability of intermediate solutions
Yassine et al. (2003)	*Rework*—global design considerations require locally acceptable solutions to be reconsidered
	Churn—"number of problems being solved (or progress being made) does not reduce (increase) monotonically"
	Convergence—"A process in which the total number of problems being solved falls below an acceptable threshold"
Bhuiyan et al. (2004)	*(New) design version*—entire design phase(s) are re-attempted following a post-phase progress review
	Churn—concurrent, functionally interdependent activities are redone in a process of "informal incremental change". Increases with the proportion of time each function spends working with the others
	Overlap spin—one, other or both of two overlapping activities are repeated if outputs are not acceptable
Cho and Eppinger (2005)	*Overlapping iteration*—an upstream task releases preliminary information to a downstream task. Iteration may be required to accommodate changes when the upstream task later releases more refined information
	Sequential iteration—a downstream task modifies information that was used earlier by an upstream task. Intermediate tasks are reconsidered in the original sequence. See also Browning and Eppinger (2002)
	Parallel iteration—tasks done concurrently with ongoing information exchange, creating work for one another according to interdependencies. See also Smith and Eppinger (1997a)
Fairley and Willshire (2005)	*Iterative prototyping*—to "help evolve a user interface"
	Agile development iterations—"closely involve a prototypical customer in a process that might repeat daily"
	Incremental build—"developers produce weekly builds of an evolving product"
	Spiral model iterations—"help the team confront and mitigate risk in an evolving product"
	Evolutionary rework—"adds value to an evolving [software] product"…"to provide new capabilities"
	Avoidable retrospective rework—modify an evolving software product to accommodate needs that developers knew about previously but did not implement, e.g. due to schedule pressure
	Avoidable corrective rework—fixing defects in a software product that were not found prior to release

Table 5 continued

References	Description of terminology
Chusilp and Jin (2006)	*Design task iteration*—tasks repeated because new information is available, or established success criteria are not satisfied
	Mental activity iteration—in the mind to "clarify problems, generate ideas, and arrive at better designs"
Taylor and Ford (2006)	*Rework*—a task is redone to implement a change, e.g. a stronger beam is installed. Leads to...
	(a) Contamination—secondary iteration caused by ripple effects after rework. For instance, replacing a beam requires the ceiling to be re-plastered. Tasks within the original project scope are revisited
	(b) New tasks—rework or contamination requires additional tasks not in original scope. For instance, the ceiling must be supported temporarily to replace the beam
	Backlog growth—ripple effects cause a tipping point where work is created faster than it can be resolved
Wynn and Eckert (2007)	*Exploration*—concurrent, iterative exploration of problem and solution during creative problem-solving
	Convergence—to find a satisficing design when parameter/objective relationships are complex
	Refinement—to improve a design without enhancing performance against primary objectives
	Rework—to correct problems that emerge as design progresses, or that are created by changes
	Negotiation—to reach trade-offs between competing goals owned by different personnel
	Repetition—to apply a similar operation to different information or different parts of the design
Jun and Suh (2008)	*Negotiation iteration*—"activities exchange design information with each other bidirectionally for obtaining a desirable design solution" ... "usually arises because of the coupled nature of product design"
	Feedback iteration—"a manager usually identifies target conformance specifications and the acceptable level of design outputs, and reviews design outputs before they are released for wider use"
	Engineering change iteration—"During the PD, design outputs are frequently changed due to technical problems"
	Refinement—due to uncertainty and complexity, a design must be iteratively refined until acceptable
	Overlap—occurs when preliminary information from an upstream task "turns out to be false or misleading"
Arundachawat et al. (2009)	*Upstream rework*—to correct faults in an upstream task that are detected downstream
	Downstream rework—iteration of a task is caused by changes to prelim. info. released by an upstream task
	Duplication—repeating similar operations on different variants of a design when doing SBCE

Text in quotations is taken from the publication indicated in the same row

agree and coordinate if stakeholders perceive the iteration differently.

One challenge is that iteration is often reflected only cursorily in process descriptions that are used in industry. Iterations may often appear to be outside the level of detail of those descriptions (Browning 1998). In the research literature, meanwhile, different modelling frameworks focus on different aspects of iteration. This may make it difficult to select the best approach to model a particular situation, especially for a modeller unfamiliar with all the issues (Wynn and Eckert 2007). Arguably, it is important to distinguish between the different iterative situations that can occur and make informed decisions about how to treat them, to create appropriate process representations and to generate understanding for management.

5.4 A characteristic form of iteration may dominate

Although the review reveals many forms of iteration, consideration of case descriptions in the literature as well as insight from our own case studies suggest that a subset of these dominate in most real-world situations. For instance, creation of engineering drawings may be dominated by rework to correct drafting errors. Integration of contributions from multiple work packages or suppliers early in a development program is often focused on dealing with updates to preliminary information. Later, iterations may focus on rework as additional operating scenarios are considered, test results are received, and production issues are worked out. Recognising the characteristic form of the iteration is arguably an important step towards handling it effectively. The taxonomy developed in this article should help to resolve differing perceptions on iteration by providing guidance to recognise and understand the dominant characteristics of a situation.

5.5 Forms of iteration may be nested hierarchically

Iterative characteristics depend on the level of resolution at which a process is examined (Wynn and Eckert 2007). For example, the processes of a design department might be characterised by certain forms of coordinative iteration when interacting with other departments; corrective iterations when considering the end-to-end processes within the department; and progressive iterations in the work of individual designers. To give another example, a concretisation process is likely to involve correcting mistakes made earlier in that process, perhaps due to information that was incomplete, imprecise, or ambiguous when decisions were made.

These examples illustrate that it may be helpful to perceive a process as different iterative situations that interact or are nested within one another. Multilevel description of a PD process according to iterative characteristics is a topic that could benefit from additional research attention.

5.6 Iteration may be positively influenced

Management of iteration can be challenging in practice (Browning 1998). One contributing factor is the complexity and ambiguity of the topic. Another is that iteration is a systemic issue influenced by many interactions on product, process, and organisational levels (Le 2012). Even if some iteration is undesirable and avoidable, people may perceive there is nothing that can be done about it from their positions in the organisation.

However, research reviewed in this article indicates that appropriate strategies and policies can influence the iterative structure of a process on many levels, both in terms of exploiting the positive effects and mitigating the negative effects. Prescriptive approaches to help manage iteration have also received much research attention, for instance based on the analysis of PD processes using DSM (Eppinger 1991), MDM (Lindemann et al. 2009), and different types of simulation (Levitt et al. 1999; Browning and Eppinger 2002; Ford and Sterman 2003; Wynn et al. 2014). There remain numerous challenges regarding the handling of iteration in those approaches, ranging from concerns about probabilistic assumptions (Smith et al. 1992) to those of avoiding deadlocks and other structural problems in a simulation (Karniel and Reich 2009). Prescriptive methods such as SBCE (Ward et al. 1995) and the design thinking process (Brown 2008) have also been developed to help practitioners manage and exploit different iterative situations. In future, we hope to review such contributions from the viewpoint of their strengths and weaknesses with respect to different situations. This could provide useful guidance for practitioners seeking to improve their processes.

5.7 Iteration in different development domains

As noted earlier, this article has focused mainly on design and product development, complemented by selected insights into iteration in the domains of software project management and construction management. We did not find any publications that provide detailed comparisons of iteration across the domains, and a thorough analysis of this issue is beyond the scope of the present article. Some preliminary comments are made in the next paragraphs.

In terms of similarities, research into micro-level iteration is usually focused on the nature of generic design activity, and insights regarding exploration and concretisation appear largely independent of domain. On the macro-level, rework is extensively discussed in all three domains. Commentaries on lean and agile may be found in all three domains, as may work on modularity and complexity. The iterative stereotypes embedded in such work, such as governance, comparison, and concentration are consequently discussed across the domains as well.

Differences are also apparent. First, Tables 1 and 6 indicate that in the literature we reviewed, many of the stereotypes and issues are not equally represented across the three development domains. Further research is needed to determine whether these differences in emphasis should be attributed to characteristics of the domains themselves, the particular contexts that were studied, and/or the interests of the research communities. A second difference is that studies in construction and software usually focus on iteration manifested in changes to an artefact, whereas research into iteration in product development typically considers work on incomplete design information. Consequently, rework in the construction sector is well documented with change orders, and this has enabled statistical retrospective analyses (e.g. Love and Edwards 2004), while researchers in software development are well positioned to analyse iteration patterns that are documented in automatic version control systems (Fairley and Willshire 2005). On the other hand, in product development projects, detailed records of iteration prior to design release are not readily available at present, and statistical retrospective analyses of iteration patterns in this domain are not common. A third difference is that influential research into macro-level iteration in product development and construction has mainly focused on undesirable forms of iteration, while research on iteration in software projects has more strongly emphasised deliberate iteration structures used to manage risk and integration in practice, for instance in agile, spiral, and other forms of incremental development.

5.8 Topics not addressed in this article

The possibility of future work to describe development processes as interacting iteration structures, to review prescriptive approaches with respect to different iterative situations, and to study iteration across development domains has already been mentioned. A number of further issues could not be addressed within the scope of this article, and these are indicated below.

Table 6 Selected key publications mapped against the taxonomy

Focus[a]	Publication	Exploration	Concretisation	Convergence	Refinement	Incr. cmpltn	Rework	New work
Micro-level empirical studies								
DS	Guindon (1990)		C					
DA	Schön and Wiggins (1992)	E						
DE	Smith et al. (1992)							
DE	Smith and Leong (1998)	E						
DE	Smith and Tjandra (1998)			O			R	
DE	Atman et al. (1999)	E	C					
DE	Adams and Atman (2000)	E						
DC	Austin et al. (2001)	E					R	
DE	Dorst and Cross (2001)	E						
DE	Ahmed et al. (2003)	E		O				
DE	Boudouh et al. (2006)			O			R	
DE	Chusilp and Jin (2006)	E						
DE	Jin and Chusilp (2006)	E						
Micro-level models								
DA	March (1984)	E						
DR	Gero (1990)	E	C					
DR	Hybs and Gero (1992)	E						
DR	Maher and Poon (1996)	E						
DR	Hatchuel and Weil (2009)	E	C					
DE	Jin and Benami (2010)	E						
	Moen and Norman (2010)							
Macro-level empirical studies								
PD	Eastman (1980)							
PD	Clark et al. (1987)							
PD	Eisenhardt and Tabrizi (1995)							
PD	Ward et al. (1995)						R	
SM	Cusumano and Selby (1997)					I		
SM	Cusumano (1997)					I		
SM	Iansiti and MacCormack (1997)							
PD	Thomke (1997)						R	
PD	Thomke (1998)			O			R	
CM	Love et al. (1999)						R	
PD	Sobek et al. (1999)						R	
PD	Terwiesch and Loch (1999)							
CM	Love and Li (2000)						R	
SM	MacCormack et al. (2001)							
CM	Love (2002)						R	
PD	Terwiesch et al. (2002)							
PD	Eckert et al. (2004)						R	
CM	Love and Edwards (2004)						R	
PD	Liker and Morgan (2006)							
SM	Dybå and Dingsøyr (2008)						R	
CM	Hwang et al. (2009)						R	
CM	Love et al. (2010)						R	
PD	Fernandes et al. (2014)							
SM	Kim et al. (2014)				F			

Table 6 continued

Focus[a]	Publication	Exploration	Concretisation	Convergence	Refinement	Incr. cmpltn	Rework	New work
Macro-level models								
PD	AitSahlia et al. (1995)						R	
PD	Ha and Porteus (1995)						R	
PD	Krishnan et al. (1995)							
PD	Krishnan et al. (1997a)							
PD	Krishnan et al. (1997b)			O				
PD	Smith and Eppinger (1997a)							
PD	Braha and Maimon (1998)		C					
PD	Loch and Terwiesch (1998)							
PD	Ahmadi and Wang (1999)						R	
PD	Hoedemaker et al. (1999)							
PD	Roemer et al. (2000)							
PD	Joglekar et al. (2001)							
PD	Loch et al. (2003)							
PD	Mihm et al. (2003)							
PD	Yassine et al. (2003)							
PD	Bhuiyan et al. (2004)						R	
PD	Huberman and Wilkinson (2005)							
PD	Braha and Bar-Yam (2007)							
PD	Schlick et al. (2013)							
Terminologies and reviews[b]								
PD	Smith and Eppinger (1993)						R	
PD	Eppinger et al. (1994)						R	
PD	Safoutin and Smith (1996)	E		O				
PD	Browning (1998)			O			R	
DE	Adams and Atman (2000)	E						
PD	Isaksson et al. (2000)			O			R	W
DE	Costa and Sobek (2003)		C			I	R	
DE	Safoutin (2003)		C			I		
PD	Yassine et al. (2003)			O			R	
PD	Bhuiyan et al. (2004)						R	
PD	Cho and Eppinger (2005)						R	
SM	Fairley and Willshire (2005)			O	F	I	R	
DE	Chusilp and Jin (2006)	E					R	
PD	Taylor and Ford (2006)						R	W
PD	Wynn and Eckert (2007)	E		O	F		R	
PD	Jun and Suh (2008)			O			R	
PD	Arundachawat et al. (2009)						R	
PD	Le (2012)	E	C	O	F		R	

Focus[a]	Publication	Churn	Governance	Negotiation	Parallelisation	Comparison	Concentration
Micro-level empirical studies							
DS	Guindon (1990)						
DA	Schön and Wiggins (1992)						
DE	Smith et al. (1992)			N			
DE	Smith and Leong (1998)		G	N			
DE	Smith and Tjandra (1998)			N			
DE	Atman et al. (1999)					S	

Table 6 continued

Focus[a]	Publication	Churn	Governance	Negotiation	Parallelisation	Comparison	Concentration
DE	Adams and Atman (2000)		G				
DC	Austin et al. (2001)					S	
DE	Dorst and Cross (2001)						
DE	Ahmed et al. (2003)						
DE	Boudouh et al. (2006)				P		
DE	Chusilp and Jin (2006)					S	
DE	Jin and Chusilp (2006)					S	
Micro-level models							
DA	March (1984)						
DR	Gero (1990)					S	
DR	Hybs and Gero (1992)					S	
DR	Maher and Poon (1996)					S	
DR	Hatchuel and Weil (2009)						
DE	Jin and Benami (2010)					S	
	Moen and Norman (2010)		G				
Macro-level empirical studies							
PD	Eastman (1980)				P		
PD	Clark et al. (1987)			N	P		
PD	Eisenhardt and Tabrizi (1995)		G				
PD	Ward et al. (1995)					S	
SM	Cusumano and Selby (1997)		G				
SM	Cusumano (1997)		G				
SM	Iansiti and MacCormack (1997)		G				
PD	Thomke (1997)		G				
PD	Thomke (1998)						
CM	Love et al. (1999)						
PD	Sobek et al. (1999)					S	
PD	Terwiesch and Loch (1999)				P		
CM	Love and Li (2000)						
SM	MacCormack et al. (2001)		G				
CM	Love (2002)						
PD	Terwiesch et al. (2002)				P	S	
PD	Eckert et al. (2004)			N			
CM	Love and Edwards (2004)						
PD	Liker and Morgan (2006)		G			S	T
SM	Dybå and Dingsøyr (2008)		G				
CM	Hwang et al. (2009)						
CM	Love et al. (2010)						
PD	Fernandes et al. (2014)			N			
SM	Kim et al. (2014)						
Macro-level models							
PD	AitSahlia et al. (1995)						
PD	Ha and Porteus (1995)		G				
PD	Krishnan et al. (1995)				P		
PD	Krishnan et al. (1997a)				P		
PD	Krishnan et al. (1997b)						
PD	Smith and Eppinger (1997a)			N			
PD	Braha and Maimon (1998)						

Table 6 continued

Focus[a]	Publication	Churn	Governance	Negotiation	Parallelisation	Comparison	Concentration
PD	Loch and Terwiesch (1998)		G		P		
PD	Ahmadi and Wang (1999)		G				
PD	Hoedemaker et al. (1999)			N			
PD	Roemer et al. (2000)				P		
PD	Joglekar et al. (2001)			N			
PD	Loch et al. (2003)	H		N			
PD	Mihm et al. (2003)	H		N			
PD	Yassine et al. (2003)	H					
PD	Bhuiyan et al. (2004)			N	P		
PD	Huberman and Wilkinson (2005)	H					
PD	Braha and Bar-Yam (2007)	H					
PD	Schlick et al. (2013)	H					
Terminologies and reviews[b]							
PD	Smith and Eppinger (1993)			N			
PD	Eppinger et al. (1994)			N			
PD	Safoutin and Smith (1996)						
PD	Browning (1998)				P		
DE	Adams and Atman (2000)		G				
PD	Isaksson et al. (2000)					S	
DE	Costa and Sobek (2003)						
DE	Safoutin (2003)		G				
PD	Yassine et al. (2003)	H					
PD	Bhuiyan et al. (2004)			N	P		
PD	Cho and Eppinger (2005)			N	P		
SM	Fairley and Willshire (2005)		G				
DE	Chusilp and Jin (2006)						
PD	Taylor and Ford (2006)	H					
PD	Wynn and Eckert (2007)			N			T
PD	Jun and Suh (2008)		G	N	P		
PD	Arundachawat et al. (2009)				P	S	
PD	Le (2012)			N	P	S	

Rows are grouped by type of study and then sequenced by publication date

[a] The focus column indicates the research area with which each publication is primarily associated, where *PD* is product development, *SM* is software project management, *CM* is construction management, *DR* is design research without a specific domain focus, *DE* is design research with an engineering focus, *DA* is design research with an architecture focus, *DC* is design research with a construction focus, *DS* is design research with a software focus

[b] Additional analysis of terminologies is provided in Table 7

Firstly, since the intention was to provide an integrative overview, space did not allow in-depth elaboration of particular stereotypes. Further work could focus on detailing individual stereotypes, perhaps through focused systematic reviews of those topics. Similarly, we did not discuss the relationships between stereotypes in detail; this is another avenue for further research.

Secondly, as explained in Sect. 1, this article has focused on iteration as it occurs within a project. Some of the reviewed publications describe iterative situations that fall outside this scope, specifically the generational learning of Eppinger et al. (1994), the macro-scale iteration of Safoutin and Smith (1996), and the evolutionary rework of Fairley and Willshire (2005). All three of these terms concern the progressive development of a design across product generations. Similarly, the incremental development of platform parts or modules for use in different designs could be viewed as iterative, as could the modification of designs already in the field. Such contexts are not explicitly emphasised in our taxonomy. Nevertheless, the (re)design processes that occur still involve iteration that could be perceived in terms of the stereotypes developed in this article.

Table 7 Aligning selected terminologies against the integrating taxonomy indicates the profusion of terms

Author(s)	Exploration	Concretisation	Convergence	Refinement	Incr. completion
Progression					
Smith and Eppinger (1993)	–	–	–	–	–
Eppinger et al. (1994)	–	–	–	–	–
Safoutin and Smith (1996)	Meso-scale	–	Micro-scale	–	–
Adams and Atman (2000)	Transformative	–	–	–	–
Isaksson et al. (2000)	–	–	Int. evolutionary	–	–
Costa and Sobek (2003)	–	Design	–	–	Behavioural
Safoutin (2003)	–	Progression	–	–	Repetition
Yassine et al. (2003)	–	–	Convergence	–	–
Bhuiyan et al. (2004)	–	–	–	–	–
Cho and Eppinger (2005)	–	–	–	–	–
Fairley and Willshire (2005)	–	–	Iterative prototyping	Refactoring	Incr. build
Chusilp and Jin (2006)	Mental	–	–	–	–
Taylor and Ford (2006)	–	–	–	–	–
Wynn and Eckert (2007)	Exploration	–	Convergence	Refinement	–
Jun and Suh (2008)	–	–	Refinement	–	–
Arundachawat et al. (2009)	–	–	–	–	–

Author(s)	Rework	New work	Churn
Correction			
Smith and Eppinger (1993)	Unexpected	–	–
Eppinger et al. (1994)	Rework	–	–
Safoutin and Smith (1996)	–	–	–
Adams and Atman (2000)	–	–	–
Isaksson et al. (2000)	Unint. repetitive	Unint. evolutionary	–
Costa and Sobek (2003)	Rework	–	–
Safoutin (2003)	–	–	–
Yassine et al. (2003)	Rework	–	Churn
Bhuiyan et al. (2004)	New design version	–	–
Cho and Eppinger (2005)	Sequential	–	–
Fairley and Willshire (2005)	Avoidable rework	–	–
Chusilp and Jin (2006)	Design task	–	–
Taylor and Ford (2006)	Rework/contamination	New tasks	Backlog growth
Wynn and Eckert (2007)	Rework	–	–
Jun and Suh (2008)	Engineering change	–	–
Arundachawat et al. (2009)	Upstream rework	–	–

Author(s)	Governance	Negotiation	Parallelisation	Comparison	Concentration
Coordination					
Smith and Eppinger (1993)	–	Expected	–	–	–
Eppinger et al. (1994)	–	Concurrent	–	–	–
Safoutin and Smith (1996)	–	–	–	–	–
Adams and Atman (2000)	Diagnostic	–	–	–	–
Isaksson et al. (2000)	–	–	–	Int. repetitive	–
Costa and Sobek (2003)	–	–	–	–	–
Safoutin (2003)	Feedback	–	–	–	–
Yassine et al. (2003)	–	–	–	–	–
Bhuiyan et al. (2004)	–	Churn	Overlap spin	–	–

Table 7 continued

Author(s)	Governance	Negotiation	Parallelisation	Comparison	Concentration
Cho and Eppinger (2005)	–	Parallel	Overlapping	–	–
Fairley and Willshire (2005)	Spiral/agile	–	–	–	–
Chusilp and Jin (2006)	–	–	–	–	–
Taylor and Ford (2006)	–	–	–	–	–
Wynn and Eckert (2007)	–	Negotiation	–	–	Repetition
Jun and Suh (2008)	Feedback	Negotiation	Overlapping	–	–
Arundachawat et al. (2009)	–	–	Downstream rework	Duplication	–

A dash indicates that a stereotype does not explicitly appear in the respective terminology

Thirdly, as noted in Sect. 5.3, iteration is an inherently subjective issue. It involves perceiving segments of activity, separated in time and occurring in different contexts, to be similar. In principle, any process perceived to involve an element of circularity could be analysed as an iteration process—although it might not be productive to do so. Stakeholders must therefore choose where to perceive iteration, and must also decide how to idealise it in descriptions and models (Dowson 1987; Isaksson et al. 2000).

Here, we have not addressed such problems of subjectivity but instead focused on analysing iterative situations as they are described in the design and development literature. It was shown that the stereotypes are sufficient to distinguish between situations as described in the literature we reviewed. A tension remains in acknowledging that iterative situations may legitimately be viewed through different lenses, while at the same time noting that significant differences are apparent e.g. when comparing exploration in early concept design to churn amongst design teams working to integrate a complex system. In this article, we do not provide guidance regarding what should and should not be considered iteration, nor do we propose a method to demarcate iterative situations encountered in practice. Another possible area for future research is to address this gap by undertaking empirical work, for instance through case study and/or survey, to investigate how different iterative situations can be identified, distinguished, and related in the field.

6 Concluding remarks

Perceiving design and development as an iteration process is necessary to understand and ultimately support it. However, despite substantial research attention, a consensus model or terminology for describing iterative situations has remained elusive. This article makes two main contributions towards resolving this issue. Firstly, insights into iteration are collected and summarised, highlighting the different research communities and research approaches through which they were developed. The second main contribution is to introduce the concept of iterative stereotypes to convey characteristics of a particular situation, to develop a taxonomy of stereotypes, and to provide a mapping from stereotypes to selected key publications in which they appear. Table 6 suggests that most prior publications are quite narrowly focused on a small subset of the iterative situations that are possible. Furthermore, the areas of literature do not all strongly cross-reference each other, especially those examining micro- and macro-level iteration and those considering the different development domains. There is a need for further research to develop more integrated perspectives on iteration—the topic is central to most analyses of design and development, suggesting that such work could yield valuable insight.

Overall, it is hoped that the review and integrating taxonomy presented in this article will provide insight to practitioners and researchers by helping to express the characteristics of iterative situations and helping to contextualise future analyses of iterative processes.

Acknowledgments The authors gratefully acknowledge John Clarkson, Nam Le, and other collaborators for discussions and feedback on the topic of design process iteration. We also thank the Editor and anonymous reviewers whose insightful comments helped to improve the manuscript.

References

Adams RS, Atman CJ (2000) Characterizing engineering student design processes: an illustration of iteration. In: Proceedings of the ASEE annual conference, ASEE

Ahmad N, Wynn DC, Clarkson PJ (2013) Change impact on a product and its redesign process: a tool for knowledge capture and reuse. Res Eng Des 24(3):219–244

Ahmadi R, Wang R (1999) Managing development risk in product design processes. Oper Res 47(2):235–246

Ahmed S, Wallace KM, Blessing LTM (2003) Understanding the differences between how novice and experienced designers approach design tasks. Res Eng Des 14(1):1–11

AitSahlia F, Johnson E, Will P (1995) Is concurrent engineering always a sensible proposition? IEEE Trans Eng Manage 42(2):166–170

Ameri F, Summers JD, Mocko GM, Porter M (2008) Engineering design complexity: an investigation of methods and measures. Res Eng Des 19(2–3):161–179

Arundachawat P, Roy R, Al-Ashaab A, Shehab E (2009) Design rework prediction in concurrent design environment: current trends and future research directions. In: Proceedings of the 19th CIRP design conference–competitive design, CIRP, pp 237–244

Asimow M (1962) Introduction to design. Prentice Hall, Englewood Cliffs

Atman C, Chimka J, Bursic K, Nachtmann H (1999) A comparison of freshman and senior engineering design processes. Des Stud 20(2):131–152

Austin S, Lyneis J, Bryant BJ (2001) Mapping the conceptual design activity of interdisciplinary teams. Des Stud 22(3):211–232

Badke-Schaub P, Gehrlicher A (2003) Patterns of decisions in design: Leaps, loops, cycles, sequences and meta-processes. In: International conference on engineering design, ICED 03, Design Society

Ballard G (2000) Positive versus negative iteration in design. In: Proceedings of the eighth annual conference of the international group for lean construction, IGLC-6, IGLC, pp 17–19

Beauregard Y, Thomson V, Bhuiyan N (2008) Lean engineering logistics: load leveling of design jobs with capacity considerations. Can Aeronaut Space J 54(2):19–30

Bhuiyan N, Gerwin D, Thomson V (2004) Simulation of the new product development process for performance improvement. Manage Sci 50(12):1690–1703

Boothroyd G (1994) Product design for manufacture and assembly. Comput Aided Des 26(7):505–520

Boudouh T, Anghel DC, Garro O (2006) Design iterations in a geographically distributed design process. In: ElMaraghy HA, ElMaraghy WH (eds) Advances in design part VII. Springer, London, pp 377–385

Braha D, Bar-Yam Y (2004a) Information flow structure in large-scale product development organizational networks. J Inf Technol 19(4):244–253

Braha D, Bar-Yam Y (2004b) Topology of large-scale engineering problem-solving networks. Phys Rev E 69(1):016,113

Braha D, Bar-Yam Y (2007) The statistical mechanics of complex product development: empirical and analytical results. Manage Sci 53(7):1127–1145

Braha D, Maimon O (1998) The measurement of a design structural and functional complexity. IEEE Trans Syst Man Cybern Part A Syst Hum 28(4):527–535

Braha D, Maimon O (2013) A mathematical theory of design: foundations, algorithms and applications, vol 17. Springer, Berlin

Braha D, Brown DC, Chakrabarti A, Dong A, Fadel G, Maier JR, Seering W, Ullman DG, Wood K (2013) Dtm at 25: essays on themes and future directions. In: ASME 2013 international design engineering technical conferences and computers and information in engineering conference. American Society of Mechanical Engineers, pp V005T06A018–V005T06A018

Brown T (2008) Design thinking. Harvard Bus Rev 86(6):84

Browning TR (1998) Modeling and analyzing cost, schedule and performance in complex system product development. PhD thesis, MIT

Browning TR, Eppinger SD (2002) Modeling impacts of process architecture on cost and schedule risk in product development. IEEE Trans Eng Manage 49(4):428–442

Bucciarelli LL (1994) Designing engineers. MIT Press, Cambridge, MA

Cho SH, Eppinger SD (2005) A simulation-based process model for managing complex design projects. IEEE Trans Eng Manage 52(3):316–328

Chusilp P, Jin Y (2006) Impact of mental iteration on concept generation. J Mech Des 128(1):14–25

Clark KB, Chew WB, Fujimoto T (1987) Product development in the world auto industry. Brookings Pap Econ Act 3:729–781

Clausing D (1994) Total quality development. ASME Press, New York

Cooper KG, Steele J, Macmillan S, Kirby P, Spence R (2002) Learning to learn, from past to future. Int J Project Manage 20:213–219

Costa R, Sobek DK (2003) Iteration in engineering design: Inherent and unavoidable or product of choices made? In: Proceedings of DETC03 ASME 2003 design engineering technical conferences and computers and information in engineering conference, ASME

Cusumano MA (1997) How microsoft makes large teams work like small teams. Sloan Manag Rev 39(1):9–20

Cusumano MA, Selby R (1997) How microsoft builds software. Commun ACM 40(6):53–61

Dorst K, Cross N (2001) Creativity in the design process: co-evolution of problem-solution. Design Stud 22(5):425–437

Dowson M (1987) Iteration in the software process; review of the 3rd international software process workshop. In: Proceedings of the 9th international conference on software engineering, IEEE Computer Society Press, pp 36–41

Dybå T, Dingsøyr T (2008) Empirical studies of agile software development: a systematic review. Inf Softw Technol 50(9):833–859

Eastman R (1980) Engineering information release prior to final design freeze. IEEE Trans Eng Manage EM–27(2):37–42

Eckert CM, Clarkson PJ (2010) Planning development processes for complex products. Res Eng Des 21(3):153–171

Eckert CM, Zanker W, Clarkson PJ (2004) Change and customisation in complex engineering domains. Res Eng Des 15(1):1–21

Eckert CM, Isaksson O, Earl CF (2014) Design margins as a key to understanding design iteration. In: Proceedings of the ASME 2014 design engineering technical conferences and computers and information in engineering conference, ASME

Eisenhardt KM, Tabrizi BN (1995) Accelerating adaptive processes: product innovation in the global computer industry. Adm Sci Q 40(1):84–110

Engel A, Reich Y (2015) Advancing architecture options theory: six industrial case studies. Syst Eng 18(4):396–414

Eppinger SD (1991) Model-based approaches to managing concurrent engineering. J Eng Des 2(4):283–290

Eppinger SD, Whitney DE, Smith RP, Gebala DA (1994) A model-based method for organizing tasks in product development. Res Eng Des 6:1–13

Evans JH (1959) Basic design concepts. J Am Soc Naval Eng 71(4):671–678

Fairley RE, Willshire MJ (2005) Iterative rework: the good, the bad, and the ugly. Computer 38(9):34–41

Fernandes J, Henriques E, Silva A, Moss M (2014) A method for imprecision management in complex product development. Res Eng Des 25(4):309–324

Flanagan T, Eckert CM, Clarkson PJ (2007) Externalizing tacit overview knowledge: a model-based approach to supporting design teams. Artif Intell Eng Des Anal Manuf 21(3):227–242

Ford DN, Sterman JD (2003) The liar's club: concealing rework in concurrent development. Concur Eng Res Appl 11(3):211–219

Galbraith JR (1974) Organization design: an information processing view. Interfaces 4(3):28–36

Gero JS (1990) Design prototypes: a knowledge representation schema for design. AI Magazine 11(4):26

Guindon R (1990) Designing the design process: exploiting opportunistic thoughts. Hum Comput Interact 5(2):305–344

Ha A, Porteus E (1995) Optimal timing of reviews in concurrent design for manufacturability. Manage Sci 41(9):1431–1447

Haller M, Lu W, Stehn L, Jansson G (2014) An indicator for superfluous iteration in offsite building design processes. Archit Eng Des Manag 11:360–375

Hatchuel A, Weil B (2009) Ck design theory: an advanced formulation. Res Eng Des 19(4):181–192

Hoedemaker G, Blackburn J, Wassenhove L (1999) Limits to concurrency. Decis Sci 30(1):1–18

Howard TJ, Culley SJ, Dekoninck E (2008) Describing the creative design process by the integration of engineering design and cognitive psychology literature. Des Stud 29(2):160–180

Huberman BA, Wilkinson DW (2005) Performance variability and project dynamics. Comput Math Organ Theory 11:307–332

Hwang BG, Thomas SR, Haas CT, Caldas CH (2009) Measuring the impact of rework on construction cost performance. J Constr Eng Manag 135(3):187–198

Hybs I, Gero JS (1992) An evolutionary process model of design. Des Stud 13(3):273–290

Iansiti M, MacCormack A (1997) Developing products on internet time. Harvard Bus Rev 75(5):108–117

Isaksson O, Keski-Seppälä S, Eppinger SD (2000) Evaluation of design process alternatives using signal flow graphs. J Eng Des 11(3):211–224

Jarratt T, Eckert CM, Caldwell N, Clarkson PJ (2011) Engineering change: an overview and perspective on the literature. Res Eng Des 22(2):103–124

Jin Y, Benami O (2010) Creative patterns and stimulation in conceptual design. Artif Intell Eng Des Anal Manuf 24(02):191–209

Jin Y, Chusilp P (2006) Study of mental iteration in different design situations. Des Stud 27(1):25–55

Joglekar N, Yassine A, Eppinger SD, Whitney DE (2001) Performance of coupled product development activities with a deadline. Manage Sci 47(12):1605–1620

Jun HB, Suh HW (2008) A modeling framework for product development process considering its characteristics. IEEE Trans Eng Manage 55(1):103–119

Karniel A, Reich Y (2009) From dsm-based planning to design process simulation: a review of process scheme logic verification issues. IEEE Trans Eng Manag 56(4):636–649

Keller R, Eckert CM, Clarkson PJ (2008) Determining component freeze order: a redesign cost perspective using simulated annealing. In: ASME 2008 international design engineering technical conferences and computers and information in engineering conference, American Society of Mechanical Engineers, pp 333–342

Kim M, Zimmermann T, Nagappan N (2014) An empirical study of refactoring challenges and benefits at microsoft. IEEE Trans Softw Eng 40(7):633–649

Kline SJ (1985) Innovation is not a linear process. Res Manag 28(4):36–45

Kolodner JL, Wills LM (1996) Powers of observation in creative design. Des Stud 17:385–416

Krishnan V, Eppinger SD, Whitney DE (1995) Accelerating product development by the exchange of preliminary product design information. J Mech Des 117(12):491–498

Krishnan V, Eppinger SD, Whitney DE (1997a) A model-based framework to overlap product development activities. Manage Sci 43(4):437–451

Krishnan V, Eppinger SD, Whitney DE (1997b) Simplifying iterations in cross-functional decision making. J Mech Des 119(12):485–493

Larman C, Basili VR (2003) Iterative and incremental development: a brief history. Computer 6:47–56

Le HN (2012) A transformation-based model integration framework to support iteration management in engineering design. PhD thesis, University of Cambridge

Lee J, Hong YS (2015) Design freeze sequencing using bayesian network framework. Ind Manag Data Syst 115(7):1204–1224

Levitt RE, Thomsen J, Christiansen TR, Kunz JC, Jin Y, Nass C (1999) Simulating project work processes and organizations: toward a micro-contingency theory of organizational design. Manage Sci 45(11):1479–1495

Liker J, Morgan J (2006) The toyota way in services: the case of lean product development. Acad Manag Perspect 20(2):5–20

Lindemann U, Maurer M, Braun T (2009) Structural complexity management: an approach for the field of product design. Springer, Berlin

Loch C, Terwiesch C (1998) Communication and uncertainty in concurrent engineering. Manage Sci 44(8):1032–1048

Loch C, Mihm J, Huchzermeier A (2003) Concurrent engineering and design oscillations in complex engineering projects. Concur Eng Res Appl 11(3):187–199

Love PED (2002) Influence of project type and procurement method on rework costs in building construction projects. J Constr Eng Manag 128(1):18–29

Love PED, Edwards DJ (2004) Forensic project management: the underlying causes of rework in construction projects. Civil Eng Environ Syst 21(3):207–228

Love PED, Li H (2000) Quantifying the causes and costs of rework in construction. Constr Manag Econ 18(4):479–490

Love PED, Mandal P, Li H (1999) Determining the causal structure of rework influences in construction. Constr Manag Econ 17(4):505–517

Love PED, Edwards DJ, Watson H, Davis P (2010) Rework in civil infrastructure projects: determination of cost predictors. J Constr Eng Manag 136(3):275–282

MacCormack A, Verganti R, Iansiti M (2001) Developing products on "internet time": the anatomy of a flexible development process. Manage Sci 47(1):133–150

Maher ML, Poon J (1996) Modeling design exploration as coevolution. Microcomput Civil Eng 11:195–209

Maier A, Störrle H (2011) What are the characteristics of engineering design processes? In: International conference on engineering design, ICED 11, Design Society

March L (1984) The logic of design. In: Cross N (ed) Developments in design methodology. Wiley, New York

Mens T, Tourwé T (2004) A survey of software refactoring. IEEE Trans Softw Eng 30(2):126–139

Mihm J, Loch C, Huchzermeier A (2003) Problem-solving oscillations in complex engineering projects. Manage Sci 46(6):733–750

Moen RD, Norman CL (2010) Circling back. Qual Prog 43(11):22

Oppenheim B (2004) Lean product development flow. Syst Eng 7(4):352–376

Osborne SM (1993) Product development cycle time characterization through modeling of process iteration. Master's thesis, Massachusetts Institute of Technology, Boston, MA

Pahl G, Beitz W (1996) Engineering design. Springer, London

Roemer T, Ahmadi R, Wang R (2000) Time-cost trade-offs in overlapped product development. Oper Res 48(6):858–865

Safoutin MJ (2003) A methodology for empirical measurement of iteration in engineering design processes. PhD thesis, University of Washington

Safoutin MJ, Smith RP (1996) The iterative component of design. In: Proceedings of international conference on IEEE Engineering and technology management, 1996. IEMC 96, pp 564–569

Schlick CM, Duckwitz S, Schneider S (2013) Project dynamics and emerging complexity. Comput Math Organ Theory 19:415–480

Schön DA, Wiggins G (1992) Kinds of seeing and their functions in designing. Des Stud 13(2):135–156

Sered Y, Reich Y (2006) Standardization and modularization driven by minimizing overall process effort. Comput Aided Des 38(5):405–416

Simon HA (1973) The structure of ill structured problems. Artif Intell 4(3–4):181–201

Smith RP, Eppinger SD (1997a) Identifying controlling features of engineering design iteration. Manage Sci 43(3):276–293

Smith RP, Eppinger SD (1997b) A predictive model of sequential iteration in engineering design. Manage Sci 43(8):1104–1120

Smith RP, Eppinger SP (1993) Characteristics and models of iteration in engineering design. MIT manuscript available online

Smith RP, Leong A (1998) An observational study of design team process: a comparison of student and professional engineers. J Mech Des 120:636–642

Smith RP, Tjandra P (1998) Experimental observation of iteration in engineering design. Res Eng Des 10(2):107–117

Smith RP, Eppinger SD, Gopal A (1992) Testing an engineering design iteration model in an experimental setting. In: Proceedings of the fourth international design theory and methodology conference, ASME

Sobek DK, Ward A, Liker J (1999) Toyota's principles of set-based concurrent engineering. Sloan Manag Rev 40(2):67–83

Suss S, Thomson V (2012) Optimal design processes under uncertainty and reciprocal dependency. J Eng Des 23(10–11):826–848

Taylor T, Ford DN (2006) Tipping point failure and robustness in single development projects. Syst Dyn Rev 22(1):51–71

Terwiesch C, Loch CH (1999) Measuring the effectiveness of overlapping product development activities. Manage Sci 45(4):455–465

Terwiesch C, Loch CH, Meyer AD (2002) Exchanging preliminary information in concurrent engineering: alternative coordination strategies. Organ Sci 13(4):402–419

Thomke S (1998) Managing experimentation in the design of new products. Manage Sci 44(6):743–762

Thomke SH (1997) The role of flexibility in the development of new products: an empirical study. Res Policy 26:105–119

Thuesen C, Hvam L (2011) Efficient on-site construction: learning points from a german platform for housing. Constr Innov 11(3):338–355

Torraco RJ (2005) Writing integrative literature reviews: guidelines and examples. Hum Resour Dev Rev 4(3):356–367

Ulrich K, Eppinger S (2011) Product design and development, 5th edn. McGraw-Hill Education, New York

Unger D, Eppinger SD (2011) Improving product development process design: a method for managing information flows, risks, and iterations. J Eng Des 22(10):689–699

Ward A, Liker J, Cristiano J, Sobek D (1995) The second toyota paradox: how delaying decisions can make better cars faster. Sloan Manag Rev 36(3):43–61

Wyatt DF, Wynn DC, Jarrett JP, Clarkson PJ (2012) Supporting product architecture design using computational design synthesis with network structure constraints. Res Eng Des 23(1):17–52

Wynn DC (2007) Model-based approaches to support process improvement in complex product development. PhD thesis, Cambridge University

Wynn DC, Clarkson PJ (2005) Models of designing. In: Clarkson PJ, Eckert CM (eds) Design process improvement: a review of current practice. Springer, London, pp 34–59

Wynn DC, Eckert CM, Clarkson PJ (2007) Modelling iteration in engineering design. In: Proceedings of the 17th international conference on engineering design, Design Society

Wynn DC, Caldwell NHM, Clarkson PJ (2014) Predicting change propagation in complex design workflows. J Mech Des 136(8):081,009–1–081,009–13

Yassine A, Braha D (2003) Complex concurrent engineering and the design structure matrix method. Concur Eng Res Appl 11(3):165–176

Yassine A, Joglekar N, Braha D, Eppinger SD, Whitney D (2003) Information hiding in product development: the design churn effect. Res Eng Des 14:145–161