A Quick Software Development Method for Improving Competitiveness of Services

Satoru Aihara, Masayuki Inoue, Akio Jin, Yoshinori Furukawa, Takako Tanaka, Nagisa Sekiguchi, Eiichi Oka, and Keitaro Horikawa

Abstract

Environmental changes surrounding our society are becoming more uncertain, and therefore, the development of services must be more flexible than ever. To adapt to such an environment while continuously receiving feedback from the market and our customers, we urgently need to switch to iterative and incremental software development. This article presents an analysis of problems concerning responses to changes in the environment, with a focus on devising effective methods for developing software. First, as an alternative to conventional software development methods, we introduce BizDevOps—an approach that enables rapid development through unified teams. Then we introduce a framework as an alternative to re-engineering, which takes several years. Specifically, we explain a re-engineering model in which multiple *One Team* groups simultaneously develop software in an iterative and incremental manner by working together with other teams while each team shares the same concept model.

Keywords: lean and agile, BizDevOps, early-phase analysis

1. Introduction: BizDevOps

We discuss in this section the concept of BizDevOps and describe ongoing efforts related to its development.

1.1 Competitiveness, speed, and company culture

Globalization and technological innovations have resulted in an environment surrounding business markets that is intensifying in terms of competition to rapidly get products to market (speed of product development). Innovative companies are shortening the time taken to develop services, whereas companies applying entrenched conventional development methods are finding it difficult to release products in an agile manner. For example, the conventional approach typically takes one year to develop new services to full specification and release them with top quality. By the time a product is released, the market may no longer need that product, and as a result, the risk of not getting timely feedback increases (**Fig. 1**).

In the vicious circle of being unable to sell products even if they are made, there is a danger that employees will lose motivation, strains will be put on the organization, and competitiveness will rapidly deteriorate. At present, many system integrators and businesses that develop their own in-house systems use the conventional waterfall approach to development, in which progress flows steadily downward through the different development phases. Although developers are frequently aware that they should adopt a faster way of doing things, it is not easy to do so while expecting to earn profits in the conventional way

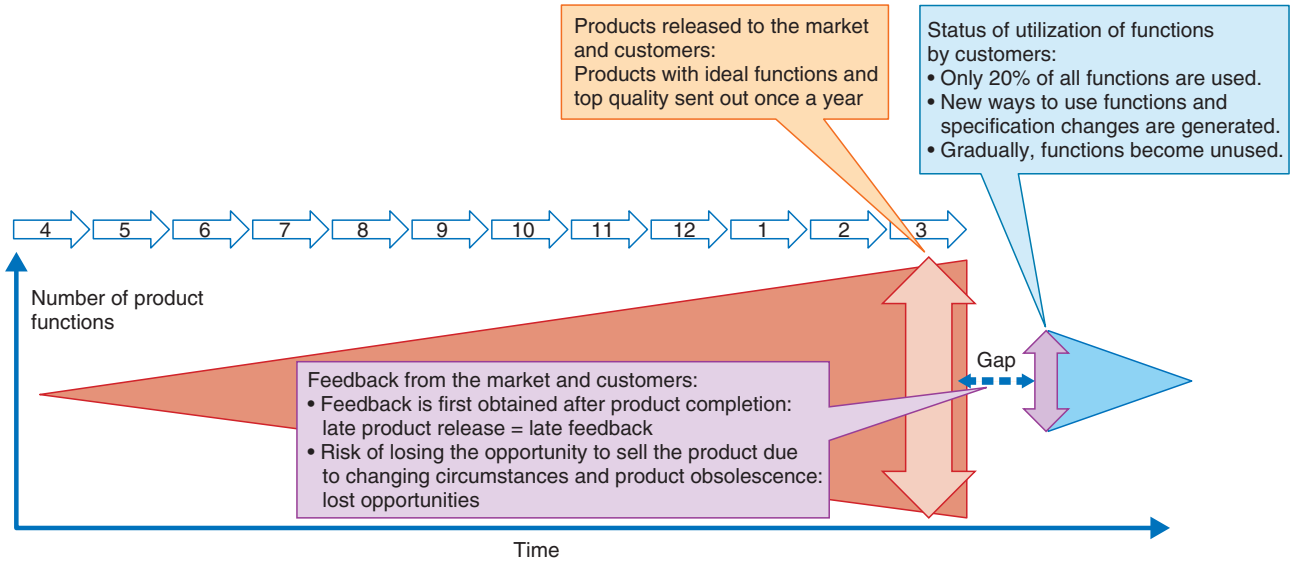


Fig. 1. Traditional service development and utilization status.

because the current situation does not alert them to a sense of impending danger. The track records of successful organizations, and their rules, culture, and customs have actually become factors that obstruct new efforts, and innovative changes are fraught with tremendous problems.

1.2 Necessity of *One Teams* sharing their thinking in a lean and agile way

The concept called BizDevOps was proposed as a means of responding to changes in the business market. The objective of BizDevOps is to foster cooperation among business departments (Biz) such as the planning and sales departments, development departments (Dev), and operations departments (Ops) in order to produce superior services that can adapt rapidly to change. We have formed an approach called *One Team*, which refers to a unified team sharing a common aim to overcome institutional hurdles such as that explained above. This approach is designed to adapt to changes in the business environment and to respond to the market quickly by identifying only functions that are actually used based on *lean thinking*, namely, omitting useless designs and implementations that go unused (such as documentation and duplicated work) (**Fig. 2**).

Typical ways to shorten the development period are to apply concepts such as continuous integration and DevOps and to utilize open source software (OSS). To further speed up product development and achieve a development speed on a par with competitors, BizDevOps can be used to regularly obtain feedback from the market as to whether or not we are heading in the right direction—ranging from the planning stage of new services to hypothesis testing of business and technical problems. Through regular cooperation between Biz and Dev, a new concept can be created, and prototypes can be implemented rapidly.

Whether this concept is correct or not can be determined by applying a way of thinking based on test marketing. Test marketing is done by releasing the product quickly onto the market (from Ops) and seeing how users respond to it and then gathering feedback. The Biz team then analyzes the feedback to determine whether the project (hypothesis) was a success or not. In addition to applying this kind of lean thinking, we must also apply *agile thinking*—namely, considering the ever-changing external environment, upgrading specifications, quickly creating concepts and operational software deemed acceptable by the market and gathering feedback about them from customers, deepening our understanding, and applying shrewd modifications—as an effective measure for team building to closely connect the Biz, Dev, and Ops parts of a One Team. As for what is more difficult about BizDevOps in comparison to DevOps, we can simply presume that it is more difficult to achieve mutual understanding among three groups with differing values than it is to do so between two groups with differing values.

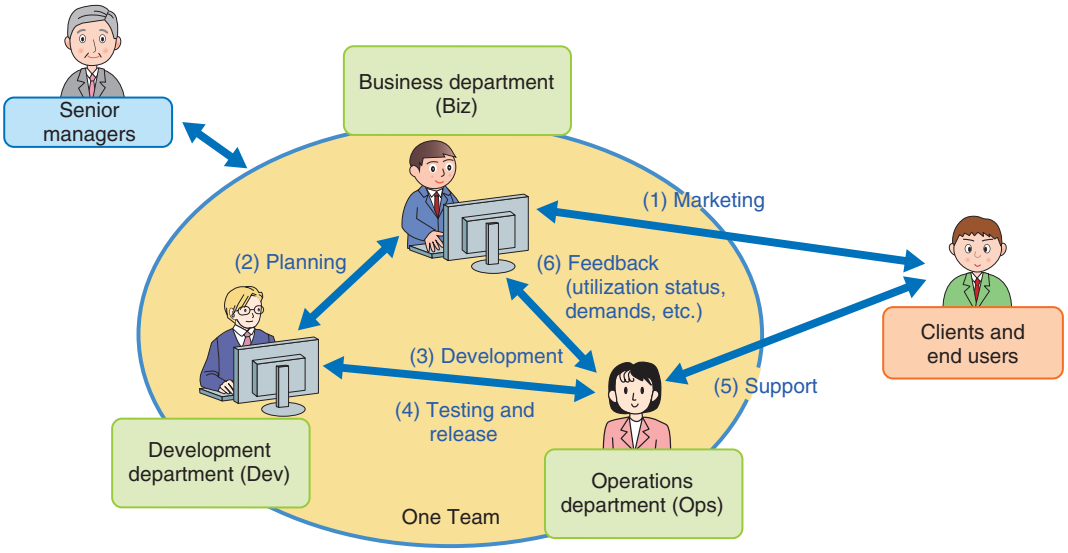


Fig. 2. BizDevOps and One Team.

1.3 BizDevOps

The BizDevOps concept is based on the Scrum way of thinking [1]. Scrum is an iterative and incremental software development approach. It involves cooperative work between people serving in three roles: the development team, the product owner, and the Scrum master. This Scrum way of thinking is extended to include the Biz, Dev, and Ops departments. To build teams that combine players with differing sets of values and cultures from the Biz component as well as the Dev and Ops components, it is essential to attain even more self-organization. Here, self-organization refers to a state in which one's best level can be autonomously mobilized under the environment in which each person is placed. It is a state under which self-motivated deeds are done naturally; everyone does their best to achieve mutual understanding and cooperation from other team members in order to accomplish the objectives of the entire team. Forming a One Team in which each member of a mixed team based on BizDevOps is self-organized makes it possible to substantially improve the success rate of projects.

We are focusing on the start-up stage of projects as the first step in adopting BizDevOps and testing and proposing effective measures for building self-organized One Teams that will have a significant effect on the success or failure of projects. While referencing self-organization and team building based on Scrum, we aim to create mutual understanding among the three representatives in each team and to rapidly change their mindset from the conventional approach (usually the waterfall method) to the new approach.

As a tangible action of this first step, we perform Scrum trials in virtual projects. By looking at how we behave when conventional methods no longer fit, how we choose tasks spontaneously (tasks are not assigned to us), how we ease friction and increase mutual understanding, and how we look back on ourselves (using the KPT (Keep, Problem, Try) method [2] framework that helps us organize and process our behavior), we analyze how we behave when we are self-organized by objectively acknowledging our behavior.

Teams that went through the Scrum trials acquired a lot of knowledge. Later on, they held workshops to teach other teams who needed to learn the knowledge they had acquired. The workshops are condensed into a maximum of ten days and involve not only typical classroom lessons and on-the-job training but also efforts to instill the mindset and skills needed for lean and agile software development. All team members actively participate in a workshop while building up and exchanging know-how in their own team. Thus, the aim of the workshops is to achieve agile team-building that drives a smooth transition to actual projects (**Fig. 3**). Moreover, the teaching team members find out more information and reach a deeper understanding through the efforts passed on to other teams; consequently, this self-organized team-building achieved through repetition can grow and expand.

For those with experience in waterfall development,

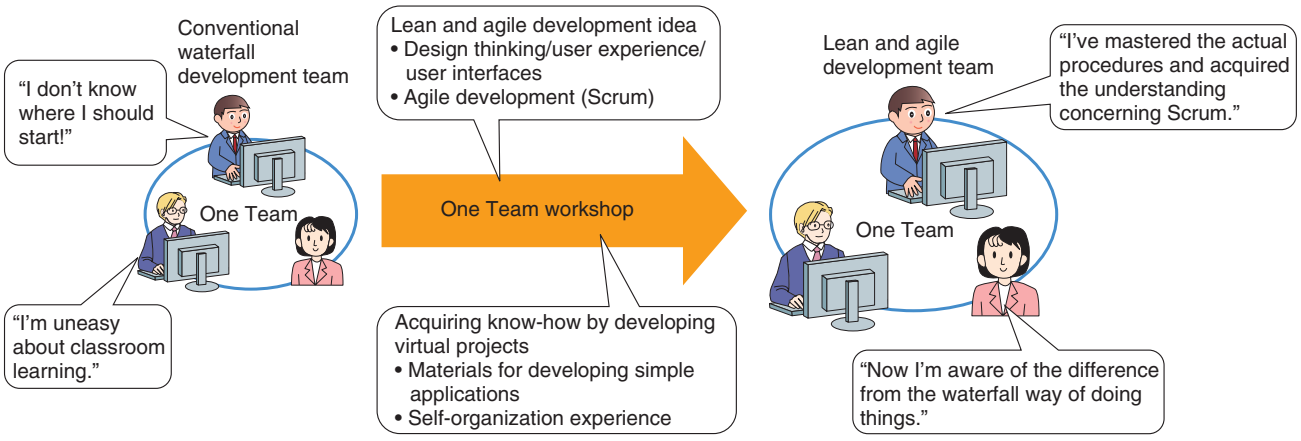


Fig. 3. Depiction of One Team workshop.

switching to a lean and agile thought process involves some confusion and resistance. However, by comparatively analyzing that different experience based on real-life experience and shifting one's thought process in accord with the essence of the know-how acquired, participants derive practical methods that help to formulate an understanding that is a great deal more agile than zero comprehension. Team members who can fully and effectively apply their strong points (including the beneficial aspects of the waterfall approach) are brought together, and methods for quickly starting up a One Team are established.

2. Applying lean and agile thinking to large-scale development

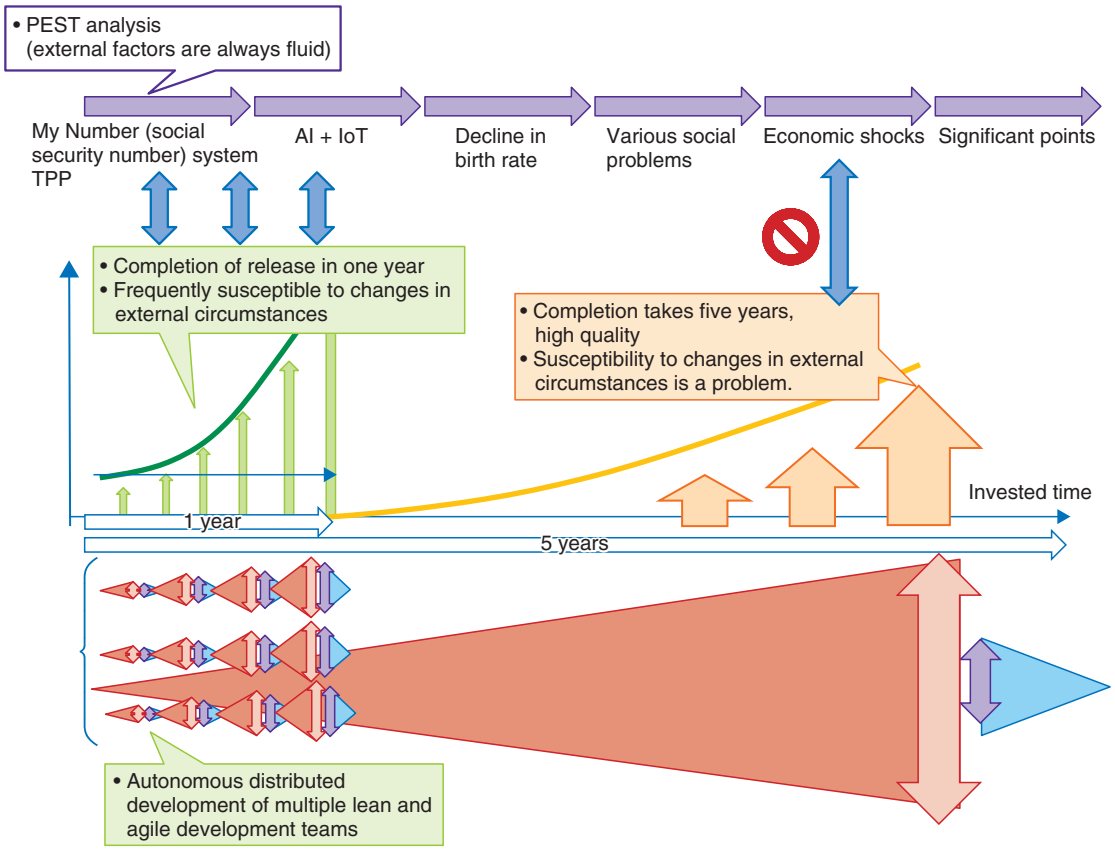
If a project manager in charge of developing an information system were given a task to complete a project that would normally take five years (estimated five years of manpower) in only one year, what methods would he/she use? Such a project would take more time to complete if the design was complicated. However, the longer it took to complete, the less users would be interested by the time the initial detailed design was developed and the product was released onto the market. It is therefore necessary to come up with measures to maintain the interest of users (**Fig. 4**). For a large-scale system to maintain its high added value, we need a new method to quickly review the system.

Hereafter, we describe a framework for linking the lean and agile way of thinking ascribed by the above-mentioned BizDevOps approach and a way of thinking called conceptual data modeling [3], which has been successfully applied to development of large-scale re-engineering.

Differing teams from the Biz, Dev, and Ops departments eliminate duplication and nonconformity between subsystems operated by each team and utilize conceptual data models to understand the *holistic* information (i.e., the big picture) needed to support the development and operation of the system after setting up One Teams. The data model used in this case is an autonomously aligned conceptual model.

2.1 Linkage of autonomously aligned conceptual model

We propose methods to implement large-scale development in an extremely short time in order to quickly adapt to changes in the external environment. In concrete terms, a system under development is divided into subsystems of suitable scale, and multiple teams (one for each subsystem) develop each subsystem in an iterative and incremental manner in parallel. Instead of devoting a long time to designing everything exactly and rigorously, self-organized One Teams are promptly set up, and each team carries out their own development autonomously (but in parallel with the others). Each small independent team develops their partial systems while receiving feedback from concerned users, and this process may speed up the software development. Related users are Biz and Ops (which evaluate and accept functions) as well as other Dev teams in related fields. Fine-tuning is necessary to make sure each subsystem conforms to the others, and efficient communication between the One Teams is essential to do this fine-tuning. Accordingly, a protocol for sharing the autonomously



AI: artificial intelligence
IoT: Internet of Things
PEST: political, economic, social, and technological
TPP: Trans-Pacific Partnership

Fig. 4. Differences between proposed method and waterfall method.

aligned conceptual model (hereafter, simply "the model") was established in order to oversee the big picture (i.e., the whole system) in a short time and plainly comprehend the work of other teams.

The model succinctly clarifies the key concepts of the whole system and the relationships between them, and it is therefore effective because it enables each team developing their own subsystem to work together to: (i) determine pointless duplication, (ii) determine complicated relationships, and (iii) specify areas to be independently managed. An example of detecting overlapping structures in data (*bad structure* detection) (**Fig. 5**) is explained in the following subsection. In general, the people who oversee an entire large-scale system are limited to a few skilled hands. Even so, the application of the model to show holistic information in this manner means that information can be shared rapidly in appropriate amounts. As a result, communication between parallel teams becomes efficient and well suited to simultaneous parallel development.

2.2 Supporting transformation from *bad* to *good* structures

Let us suppose that the whole system is separated into n individual regions (subsystems and development teams). Ordinarily, there are many dependence relationships between n subsystems. In the course of autonomous development, it is likely that interfaces (IFs) between subsystems will be frequently modified. It is thus necessary to pay attention to such modifications, thereby placing a considerable load on each development project. Accordingly, elements that require monitoring of changes to IFs between systems and maintenance of consistency are given over to tools. These tools provide overviews of an entire

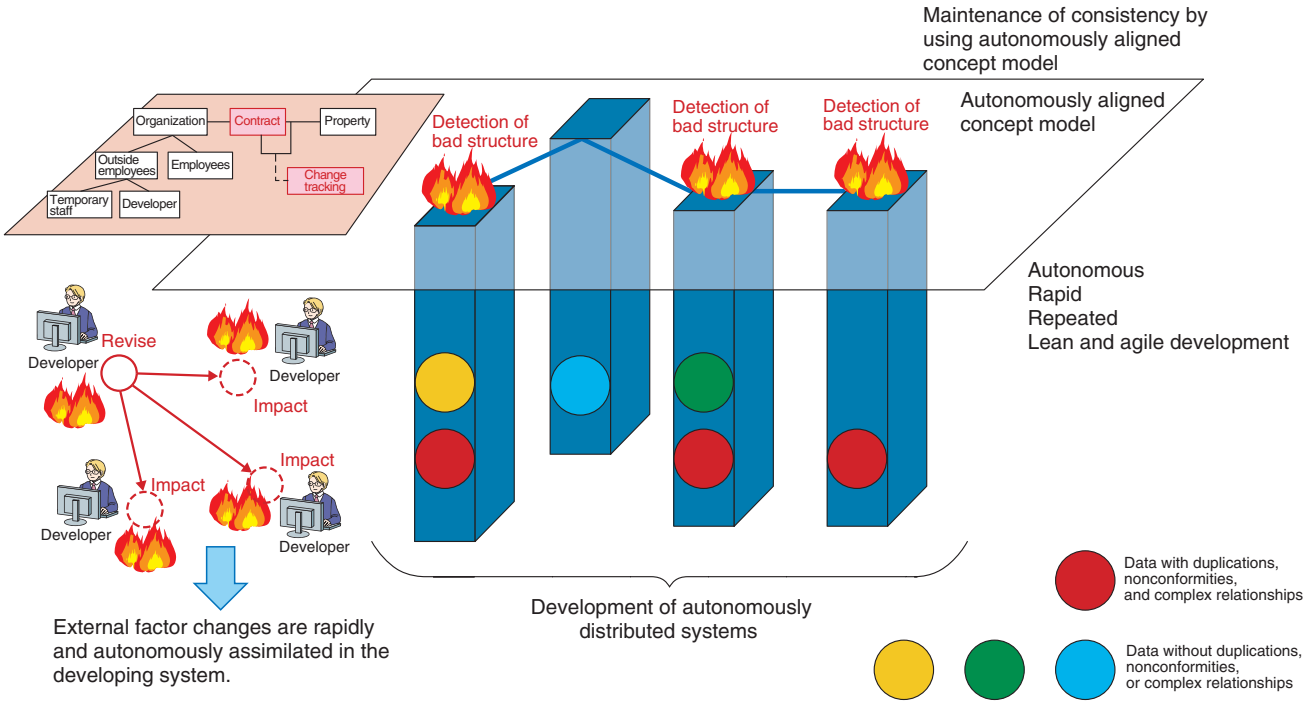


Fig. 5. Lean and agile development utilizing autonomously aligned concept model.

development project for each development team, monitor nonconformities (bad structures) between divided subsystems, and support transformations to consistent states (good structures).

Examples of bad structures are relationships involving duplication of concept model structures, nonconformities, discrepancies, and complications, as shown in **Fig. 6(a)**. In contrast, a concrete example of a good structure is a simply organized structure that is compatible with the concept model, as shown in **Fig. 6(b)**. We prototyped a function for supporting transformation to good structures by automatically detecting bad structures from this overview as a design tool (Fig. 5). By reducing workloads by having the developer orchestrate workloads between workers, each team can concentrate on the development work in their autonomous region, which will improve work efficiency.

Moreover, before and after developing the model transformation of bad structures to good ones, we prototyped a function for estimating initial running costs and evaluating designs. This function makes it possible to rapidly spot nonconformities between data models of the interior and exterior of subsystems and to continuously maintain conformity of the whole system while carrying out the parallel development of each subsystem in a lean and agile manner. The reduction in the initial running costs before and after eliminating nonconformities can be estimated, and that estimate can be effectively utilized as information necessary for making decisions.

3. Concluding remarks

BizDevOps—a development approach used by businesses for analyzing their adaptability to changes in the external environments and for speeding up development through the establishment of One Teams—was described in comparison with the conventional approach. Furthermore, we proposed a method for rapidly coordinating development teams while receiving feedback from the external environment and customers as an alternative to re-engineering for traditional large-scale systems, which requires a long development time.

The main purposes of this proposal are threefold: (i) to increase the speed of development to rapidly adapt to changes in external environments; (ii) to broaden methods for detecting internal and external changes and accommodating those changes into the development scheme; and (iii) to rapidly create services based on the detection results and strengthen

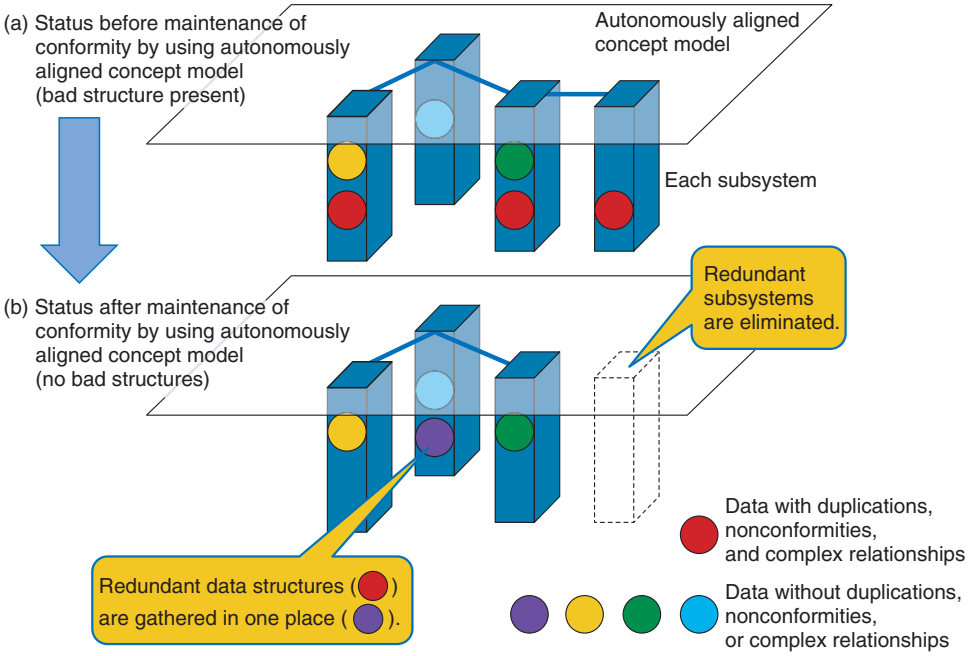


Fig. 6. Transforming faulty structures to ideal structures that can be detected.

competitiveness. As a means of combining a divide-and-rule way of thinking with lean and agile thinking, a development method with which One Teams utilize an autonomously aligned concept model for adjusting overall conformity was devised.

A tool for automatically detecting bad structures and supporting transformation to good structures is being prototyped in order to alleviate operational adjustments to changes in model structures done manually. Furthermore, we will continue to experiment with and evaluate the proposed method as a new development approach that can easily maintain overall conformity while utilizing a cost-estimation function and enabling autonomous and simultaneous work in parallel.

References

[1] Scrum guides, http://www.scrumguides.org/download.html
[2] S. Amano, "KPT Is the Answer!", Subarusya Linkage Corporation, 2013.
[3] NTT Information Network Laboratory Group, "Database Conceptual Design," Abe Publishing Ltd., 1993.

Satoru Aihara
Senior Research Engineer, Software Engineering Project, NTT Software Innovation Center.

He received a B.E. in information technology from Yamaguchi University in 1992. He joined NTT DATA in 1992. His work has focused on the development of search engine and knowledge management systems. His current research interests include software development processes.

Masayuki Inoue
Senior Research Engineer, Software Engineering Project, NTT Software Innovation Center.

He received a B.E. and M.E. in electrical engineering from Tokyo University of Science in 1994 and 1996, and a Ph.D. in engineering from Tokyo Institute of Technology in 2008. He joined NTT in 1996 and worked on the development of 3D cyberspace and video communication systems. He is currently working on system analysis in early phases using stored data in legacy systems. His research interests also include user interaction and video system evaluation. Dr. Inoue is a member of the Institute of Electronics, Information and Communication Engineers (IEICE).

Akio Jin
Senior Research Engineer, Software Engineering Project, NTT Software Innovation Center.

He received a B.E. and M.E. from Nagoya University, Aichi, in 1992 and 1994, and a Ph.D. from the University of Tsukuba in 2008. Since joining NTT in 1994, he has contributed to the development of the MPEG-4 TwinVQ audio codec, a speech recognition system, and an automatic program generation system. His current research interests include upper software development processes and structure estimation of existing systems. He received the Nikkei BP Technology Award from Nikkei Business Publications, Inc., Japan, in 1997 and the Telecom System Technology Award from the Telecommunications Advancement Foundation, Japan, in 1998.

Yoshinori Furukawa
Senior Research Engineer, Software Engineering Project, NTT Software Innovation Center.

He received a B.E. and M.E. from Ritsumeikan University, Shiga, in 1994 and 1996. He joined NTT in 2014. His current interests include software engineering.

Takako Tanaka
Research Engineer, Software Engineering Project, NTT Software Innovation Center.

She received a B.M.I.S in information science from Sanno University, Kanagawa, in 1998. She joined NTT Software Corporation in 1998 as a software engineer. Her current interests include software engineering.

Nagisa Sekiguchi
Software Engineering Project, NTT Software Innovation Center.

He received a B.E. and M.E. from Yokohama National University, Kanagawa, in 2014 and 2016. He joined NTT in 2016. His current interests include software engineering. He is a member of the Association for Computing Machinery.

Eiichi Oka
Senior Research Engineer, Supervisor, Software Engineering Project, NTT Software Innovation Center.

He received a B.S. and M.S. in nuclear engineering from Hokkaido University in 1987 and 1989. He joined NTT in 1989. He has worked on the development of telecommunication services including PHS (personal handy-phone system), 2G/3G-mobile, and the Next Generation Network. His current research interests include software engineering. He is a member of IEICE.

Keitaro Horikawa
Senior Research Engineer, Supervisor, Software Engineering Project, NTT Software Innovation Center.

He received a B.S. and M.S. in information engineering from Niigata University in 1988 and 1990. He joined NTT in 1990. He received the Best Paper Award for young researchers at the Information Processing Society of Japan national convention in 1995. He has a Project Management Professional certification from the Project Management Institute and a TOGAF (The Open Group Architecture Framework) 9 certification from the Open Group. His research interests include software design and architecture, concurrent distributed object computing, meta-object programming and computational reflection, single sign-on for web services, lightweight programming languages, mobile cloud computing and big data processing, machine learning open source projects, computer supported cooperative work, computer supported collaborative learning, game theoretic modeling for decision making, and human factors in research and development.