

Assignment

by Xu Yang

Submission date: 15-Dec-2022 02:04PM (UTC+0000)

Submission ID: 192541448

File name: 40849_Xu_Yang_Assignment_460463_940609837.docx (661.36K)

Word count: 5412

Character count: 28625

Assignment

Xu Yang

University of Wales Trinity Saint David
2200550@student.uwtsd.ac.uk

Abstract—Nowadays, the requirements for software functions and quality are constantly increasing, and how to quickly deliver high-quality software products has become the top priority of enterprises. In the situation where the traditional development model can no longer meet the needs, iterative and incremental agile development has become more and more popular. As a classic framework of agile development, scrum has become the first choice of many enterprises. It embodies the core essence of agile development and can ensure the punctuality of delivery through the accurate evaluation of user story and the use of burndown charts. In addition, DevOps has become a hot concept in recent years. As one of the classic tools of Software Development Operation, Ansible can efficiently replace human to complete repetitive tasks to improve efficiency and reduce errors.

Keywords—Agile; Scrum; user story; user story map; burndown chart; Ansible

I. INTRODUCTION

With the development of society and the improvement of computer technology, the scale of software systems is constantly expanding, and the demands for software quality are also increasing rapidly. For enterprises, how to deliver products with low cost and high quality on time has become the focus. Within this context, the traditional waterfall development model is gradually unable to meet the needs of enterprises. The linear standard waterfall model can efficiently complete the development of some standardized projects, but the delivery cycle is long and can not adapt to the variability of requirements.

Agile development emerges as the times require, which is known as an alternative to the traditional software process[1]. The concept of agile project management originated in the United States in the 21st century and is used to compare with the waterfall model. In 2001, the concept of agile was formally proposed in Utah, USA. Kent Beck and other 16 software industry experts established the Agile Alliance and signed the Agile Development Manifesto. As of today, agile development has gradually replaced the waterfall development model and has become the most popular software development method today.

Agile development adopts a human-centered, iterative, and incremental software development method, which can adapt to changing software development needs and deliver high-quality working software quickly. It has 4 essential concepts: multiple short iterations, emphasis on face-to-face communication and collaboration, adaptability to changing requirements and quality of ownership throughout the product cycle. The agile approach prioritizes interpersonal interaction and teamwork, reacts fast to changes in needs through quick

iterations and step-by-step procedures, and successfully integrates technology and business. The phrase "agile method" refers to a new kind of software development approach rather than a particular development process or methodology. Scrum mode, XP mode, etc. all belong to agile development[2].

Scrum has gained its popularity in recent years[3]. Capers Jones believes that the Scrum method can increase software productivity by 6 times from the current situation. As one of the core frameworks of agile development, scrum achieves the four important principles of agile development. It continuously seizes and adapts to changing needs by using user story and user story mapping, reasonably estimating and adjusting the development velocity in the iterative development cycle, working with exquisite teamwork, and finally achieve a high-quality iterative development process.

II. SCRUM METHODOLOGY

The term Scrum originated in the game of rugby as short for scrummage or scrumage. In 1986, Hirotaka Takeuchi and Ikujiro Nonaka proposed the application of Scrum to product development for the first time. In 2001, the Agile Alliance was established, the Agile Manifesto and Agile Principles were issued, and announced that Scrum is one of the agile methods.

Scrum is a method for developing software progressively in a complicated setting. Scrum uses an empirical process, control and permits development to advance as close to the brink of chaos as the development organization can bear allows imperfection to occur, and completes the entire product through continuous iteration and integration[4]. Scrum agile development realizes the management process of incremental iterations of software projects, which can be applied to the development and maintenance of software projects at the same time. It is not a standardized process, but a process framework in which specific roles, activities, events and corresponding rules are used to ensure the smooth progress of the software development process.

To put it simply, the Scrum framework is mainly composed of three major roles, three artifacts, five values and four activities.

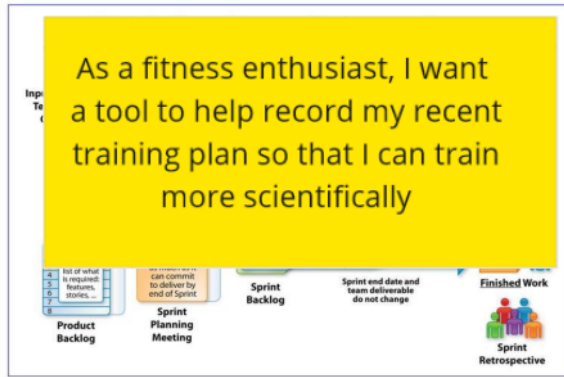


Fig. 1. Scrum Framework

As one of the most popular agile development framework[2], Scrum deeply embodies the four major aspects of agile development characteristics.

A. Multiple iterations

In agile development, frequent delivery is the core, and each iteration only needs to achieve the goal. In Scrum, the project mainly iterates around sprints, and each sprint lasts for 2-4 weeks[5]. The product requirements will be recorded in the product backlog, and each item will be represented by a user story. Each time, the most important stories for users will be selected for development and added to the sprint backlog. At the end of the sprint, the deliverable product increments are handed over.

Product backlog, sprint backlog and increment are the three major artifacts of Scrum. The use and implementation of the three artifacts play a vital role in the success of Scrum.

Because sprint is the core of scrum, as the source of sprint, the creation of user story is very important. User story is a common term in agile development, to tell the Scrum team what user really wants, which consists of 3 components: who wants it, what they want and why. A story should be described in concise and clear language, and technical details should be ignored for the time being. Scrum teams need to try their best to make a good story to ensure the realization of value and output good results.

In agile development, people break down the large bodies of work (called epics) into a number of smaller tasks (called story). In the same way, the scrum team create product backlog from the demands of customers, stakeholders and etc., using sprint to finish every stories, and finally achieve the whole project. The establishment, modification and addition of stories are inseparable from the cooperation of team members. User stories are an excellent tool that can carry items of customer or user value throughout Scrum's value creation process. However, if the stories are all the same size, it can be difficult to plan high-level and get the benefits of detailing. Story maps are for breaking down big stories as you tell them.

Fig. 2. Good user story

A user story map is a very large user storyboard on which all user stories to be developed are listed. It can be arranged horizontally and vertically according to the user's importance, priority, and module segmentation. The ultimate goal is to completely allow a user to complete a product experience through the story map. Just like finding a destination based on a map, the user story map is a similar concept.

The user's product experience is sometimes difficult to verify only by imagination. Through the user story map, this information can be displayed intuitively, and it can be imagined that a single user story is a bunch of scattered branches and leaves. Through the logical relationship between the stories, these leaves are connected to form a complete story tree. Content under a module is also easily converted into a release plan. This is also one of the highlights of user story maps. User stories for module content can be completed after one or more substitutions to form a releasable version. Then post it before advancing the map to the next story module.



Fig. 3. user story map

B. Emphasis on face-to-face communication and collaboration

Agile methods focus on the role of people, and believe that projects are completed by people, not processes or tools. It emphasizes team value and team communication and collaboration, and the scrum team realizes this concept.

In the traditional way of working, the development team will have many different roles. However, in the Scrum way of working, there are only three roles in total, and these three roles are the Product Owner (PO), Scrum Master and the Development Team. The team size is usually 4 to 10[6], and they work together to overcome difficulties[7].

The scrum team works around the sprint, which contains 4 core Ceremonies: Sprint Planning, Daily Standup, Sprint Review and Sprint Retrospective[8].

The Sprint planning meeting is a meeting used to formulate the work plan to be done in the next Sprint, and this work plan is completed by the entire Scrum team[9]. Team members refine the user stories included in this sprint, split tasks, prioritize, and analyze test points in sprint planning meeting. Specifically do the following:

- The PO and the team discuss the background of the user story, business goals, user roles, scenarios, business processes, rules, etc., to ensure that the team fully understands it.
- PO discusses the interface and interaction process with the team, and outputs wireframes or prototypes.
- The PO and the team discuss the test points of the user story, the technical implementation plan, and the possible technical risks, and must output the test points.
- The team estimates the size of the user story (story points), and splits too large user stories into smaller ones. The initial estimation is carried out by PO and SM, then by SM and developers, and organizes testers to estimate the test scale, and finally integrates centrally.
- PO prioritizes user stories. The priority only needs to be decided by the PO, and other people's opinions should be put forward in the early stage. Here, the PO is to make the final summary decision.

Scrum teams can also strengthen communication and cooperation between members through daily stand-up meetings, which is a small discussion meeting at the same time and place every day of the Sprint, limited to 15 minutes. At the meeting, all development members have to make short speeches, summarize and describe yesterday's work content and problems encountered, and formulate a work plan for the next 24 hours. Through the daily stand-up meeting, the development team can check the progress of the Sprint, discover problems in time and find solutions to the problems, meeting focuses on how to do better in the next Sprint.

The Sprint review meeting is held when a Sprint is about to end. The goal of the meeting is to inspect the delivered product increments and show the progress, adjust the product backlog accordingly and predict the end time of the Sprint[10].

C. Adaptability to changing requirements

Agile development, just like its name, is highly adaptable and flexible. Its adaptability is mainly reflected in three aspects as follow:

- Reduce the time it takes to submit a project to a user.
- Increase communication between users, employees, and developers.
- Reduce the cost of reworking to increase the software's adaptability.

In Scrum, inspection and adaptation can be carried out through three activities:

- The daily routine meeting evaluates how the Sprint goal is coming along and makes necessary adjustments to maximize the value of the work for the following day.
- The Sprint review and planning meeting evaluates the release goal's progress and makes necessary revisions to maximize the work value of the following Sprint.

- The Sprint Retrospective Meeting is intended to assess the finished Sprint and identify any enhancements that could make the following Sprint more effective and fulfilling.

The Scrum methodology is built on the philosophy of empirical process control. Three elements make up the pillar of the Scrum process framework: Transparency, Inspection, and Adaptation.

The term "transparent" means to upholding a high level of visibility in all areas of the software development process as well as upholding transparency in all areas that have an impact on the deliverables for everyone involved in their dAll aspects of the development process must be inspected frequently enough to ensure that major deviations in the process can be found in time. Another aspect is to test the skill level and motivation of the people working on the product. Inspections in Scrum can be achieved through scrum activities

The method or contents should be modified if the inspector determines that one or more process elements do not satisfy the acceptance standards and the final result is subpar. To prevent further deviations, the modification must be put into place as soon as possible. In Scrum, there is always a formal opportunity to evaluate and modify. These occasions have been planned precisely to guarantee openness and visibility. Scrum's guiding philosophy is "Inspect and Adapt." If something is done well, consider why, and then consider how to make it better.

Scrum also uses an evolutionary design to progressively arrive at the best technical solution. The team will be encouraged to find the optimal technical solution by ongoing reworking in accordance with the established needs, and won't hastily make important but incorrect technical decisions owing to incomplete knowledge. the significant risks and expenses associated with later large-scale restructuring. As a result, from the standpoint of technical execution, this can assist the team in accepting changes more readily.

D. Quality of ownership throughout the product cycle

Agile development's fundamental principles emphasize producing things rapidly and in short iterations, hence projects typically involve several iterations. As a result, each sprint's production of quality data can match to the quality of the project process. The project's quality trend may be recognized after several iterations. If there are multiple projects in progress in the same period, the quality of each project can be compared through the comparison of quality data, and at the same time, it can be visually analyzed and compared with charts.

In the traditional development model, the testing work is generally at the end of the project, which makes it possible for quality problems to be exposed very late. In the scrum framework, in order to ensure the quality of delivery, the team needs to make unified requirements on the definition of completion, the standard of completion or the criterion of completion, which is the birth of DoD. The full name of DoD is Definition of Done, which is often called "definition of done" in Agile. DoD has several major characteristics:

DoD is the completion criterion, which means that there is no need to do anything else, and it can be delivered directly.

DoD is 100% complete, not 99% or 95%.

DoD defines the minimum set of activities to achieve the goal, non-value-added, useless activities are not included in this list.

DoD is the product quality activity standard, which represents the team's consensus and commitment to quality investment in order to deliver quality.

DoD ensured a minimum quality range while unfinished business became clear. In scrum, quality is shared by the entire team. Different from the traditional development model, scrum makes the test work as much as possible a part of DoD, and tests as early as possible in each sprint to avoid problems left to the final stage.

Requirements Clarification is a process of ongoing communication between the PO and the team, which we sometimes refer to as the "three-way handshake." For the first handshake, for instance, a customer's demands are merely expressed as a carrier or container (Card) in the form of a user narrative, but a user story is not a requirement specification, and the PO and the team have a face-to-face conversation to grasp the requirements (Conversation). Then, note the requirements' scope from the users' and business' perspectives (Confirmation). Typically, the initial greeting takes place at the meeting to groom the product backlog.

The second handshake happened at the Sprint planning meeting. The PO may have a new understanding and update of the user story communicated last time. At the same time, the team will also ask some questions, so Scrum requires the PO to attend the Sprint planning meeting.

Even though we have fully communicated during the two handshakes, it is not ruled out that members of the development team may have doubts about the acceptance criteria during the specific implementation process. This is what we call the third handshake.

With at least three face-to-face conversations and Handshake, any one demand will not be too biased. This is very different from the traditional way of communication that relies heavily on documents.

III. ESTIMATING OF STORY & DURNDOWNCHART

When a team develops a project, it always needs to know when it will be completed, or how far it can be completed at a certain point in time. Like the traditional development model, before the work starts, the workload of what a team should do need to be evaluated. One difficult we face is deciding the size or complexity of a user story. Compared with traditional workload estimation methods, agile estimation has the following characteristics:

1. Collective team estimate

The agile team uses a communal estimation approach for workload estimation because in the Scrum development process, the team shares responsibility and collectively commits to the work of each sprint. Collective estimation usually uses planning poker as a tool, where the team estimates collectively by playing an estimation game. Using planning poker to do workload estimation is the most effective and very interesting way of estimating. Planning poker consists of a set of numbers similar to the Fibonacci sequence, these numbers

include: 1,2,3,5,8,13,21,etc, and each poker has four sets of such numbers for 4 people. The steps to use Planning Poker are as follows:

- Step 1: Each team member gets a set of cards, including 1,2,3,5,8,13,21,etc, a total of 12 cards.
- Step 2: The product owner or a team member plays the role of the reader. He is responsible for reading the items that need to estimate the product Backlog and asks everyone if they have any questions.
- Step 3: The team discusses this item.
- Step 4: After the team understands this entry, each team member gives an estimation result according to their own ideas, and chooses the corresponding poker card to play. The estimation result cannot be told to others, and the number is buckled on the table face down when the card is played.
- Step 5: After everyone has played their cards, the reader confirms to everyone whether the estimated results have been determined. After confirmation, count "1, 2, 3", and everyone displays the estimated results at the same time.
- Step 6: The team evaluates different estimates. Are we on the same page? Do we have differences? Is there something I haven't considered? After the discussion, there can be another round of estimation, and finally the team needs to reach a consensus.
- Step 7: Return to step two and begin estimating the following item.

2. Estimate size instead of time period, use relative estimation instead of absolute estimation.

The time it takes for people with different abilities to complete the same job is obviously different. If a person wants to estimate the absolute distance in kilometers from his/her home to his/her company, he/she may not necessarily know it, but if he/she goes to work by subway, he/she must be easy to know how many stations are there from his/her home to the company to roughly know how long the journey will take. In agile estimation, the absolute time and period are not estimated, but the size and relative value. Agile estimation uses story points as a unit of measurement, which is a multiple. Generally, the size of a function considered to be the smallest is used as a reference benchmark, which is defined as 1 story point, and other stories are compared with it. If it is 2 double the size, that's 2 story points, and if it's 5 times the size, it's 5 story points.

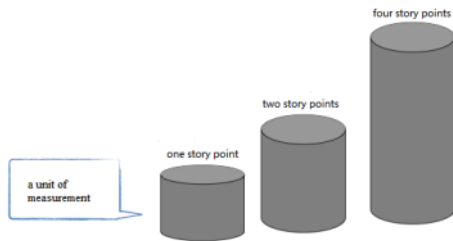


Fig. 4. Story point

3. Record team velocity for each Sprint

The sprint velocity is the number of story points that a Scrum team actually completes in a Sprint, it can tell how fast the team is doing.

There is no initial velocity for newly formed teams, products, or projects. We must keep track of each Sprint's speed so that it may be used as a benchmark for future plans. One or two Sprints can be used to determine the beginning speed throughout the execution of the Sprint.

You may calculate how many Sprints you'll need to accomplish to reach the cycle by determining all the story points in the product backlog and then knowing the average speed of each Sprint team.

Scrum also uses burndown charts to track the rest of the work, which is a way to predict when the project will be completed. Early warning system allowing allow scrum teams to see if they will miss our deadlines. Each sprint has a unique burndown chart, which may be readily made on Excel or a whiteboard. The X axis and Y axis are the two main value sections on the burndown chart. The sprint's working days are represented by the X axis, and the remaining workload is shown by the Y axis. The Scrum Team will note and estimate the specific tasks that must be finished within this Sprint at the start of the cycle. The cumulative workload is made up of all jobs that need to be finished within the Sprint but have not yet been finished. The team will update the cumulative workload each day in accordance with the progress. The sprint will be deemed successful if the total workload drops to zero at the end of it. At the beginning of Sprint, the increased workload may be greater than the completed workload, So the Burndown chart may show a slight upward trend.



Fig. 5. Burndown chart

IV. RESEARCH ON ANSIBLE

Combining the words "development" and "operations," the term "DevOps" refers to a collection of procedures, techniques, and tools that are used to foster coordination, cooperation, and integration between the departments of development (application and software engineering), technical operation (operation and maintenance), and quality assurance (testing). Developers and operation and maintenance staff have increasingly split into two divisions as a result of the complexity of business and the expansion of the workforce. There is a chasm between them because of the separation of their workplaces, the differences in their toolkits, and the differences in their business goals. Software publishing involves transferring a complex piece of code from one side of a gap to the other.

Software developers (Dev) and IT operations technicians (Ops) are encouraged to communicate with one another and work together as part of the DevOps culture, movement, or practice[11]. Building, testing, and deploying software can be done more quickly, often, and reliably by automating the "software delivery" and "architecture change" processes. Testing and software development/deployment are automated using DevOps tools[12]. Ansible is one of the many DevOps tools that are available on the market.

What is Ansible? It is a tool to help developers automate IT tasks. The development process often encounters some trivial and repetitive IT tasks, at this time, Ansible can be used instead of human execution. In a complex IT infrastructure with dozens of servers, manually performing some IT tasks may be very time-consuming. For example, if there are 10 servers, and distributed services are running on these 10 servers. If a new version of this software has been released and needs to be deployed to all servers, or when someone wants to update the Docker version on these 10 servers, sometimes this task may involve multiple steps, such as changing the configuration, restarting process, etc. A system administrator needs to repeat it on each server and it takes a lot of time to do it manually. Such as restarting processes, creating scheduled task, creating users, assigning user groups, granting permissions, etc. In all these scenarios, the method of using manual operations is to use SSH to connect to the first server, perform some operations to complete the tasks and verify them. After evaluating, connect to the next server to do the same. This manual execution is undoubtedly painful and time-consuming, especially when

multiple steps are involved and you need to remember what you did in the previous step.

Ansible can make these scenarios more efficient and waste less time. There are 4 different ways:

1. You can perform IT tasks on your own server without connecting all servers one by one via SSH.

2. Operations such as configuration, installation, and deployment can be included in a single YAML file without manually executing shell scripts.

3. The same file can be reused multiple times in different environments without manual operation.

4. When performing complex IT tasks, using tools such as Ansible to automate execution will be more reliable and make less errors.

Ansible supports all infrastructures, from operating systems to cloud services. It can work on virtual cloud hosts and physical servers, so users do not need to pay attention to infrastructure but only to specific IT tasks. All target nodes can be accessed through SSH on the user's control node. This is one of the unique advantages of Ansible. Ansible is agentless. It does not need to install anything on the target node, just install it on the control node. A control node can even be a laptop, and this machine can manage many target nodes. This reduces the deployment pressure in the initial stage, and when Ansible is updated, only the control node needs to be updated without paying attention to the target node.

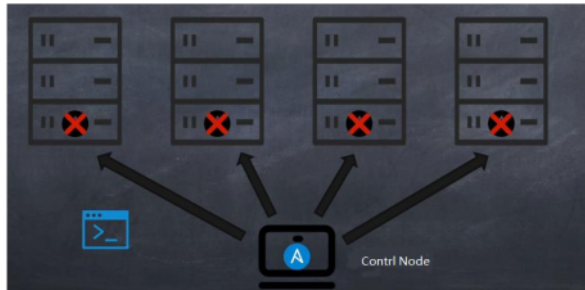


Fig. 6. Ansible operation

How does Ansible work? It uses modules to work, modules are small programs that actually do tasks, they are sent from the control node to the target node, and then the work is done on the target node, such as installing a service, stopping the process, applying a firewall, etc., which will be removed after completing the task. Modules are fine-grained, and one module can complete a specific task. Ansible has hundreds of modules, each module can perform a specific task, all IT tasks can be performed in Ansible.

All modules

- `aci_server` - Manage AIO Networks ACI/Leaf/Spine/Thunder/Thunder device/ server object
- `aci_server_group` - Manage AIO Networks ACI/Leaf/Spine/Thunder/Thunder device/ server group
- `aci_virtual_server` - Manage AIO Networks ACI/Leaf/Spine/Thunder/Thunder device/ virtual servers
- `aci_user` - Manage AAA users (aaaUser)
- `aci_user_certificate` - Manage AAA user certificates (aaaUserCert)
- `aci_access_port_block_is_access_port` - Manage port blocks of Fabric interface policy leaf profile interface selectors (infraPorts, infraPortBk)
- `aci_access_port_block_is_interface_policy_leaf_profile` - Manage Fabric interface policy leaf profile interface selectors (infraPorts, infraPortBk, infraPortBkGrp, infraPortBkGrpGrp)
- `aci_access_sub_port_block_is_access_port` - Manage sub port blocks of Fabric interface policy leaf profile interface selectors (infraPorts, infraPortBk)
- `aci_aep` - Manage attachable Access Entity Profile (AEP) objects (infraAEP, infraAEPGrp)
- `aci_aep_is_domain` - Bind AEPs to Physical or Virtual Domains (infraAEPGrp)
- `aci_app` - Manage top level Application Profiles (APP) objects (infraApp)
- `aci_bd` - Manage Bridge Domains (BD) objects (infraBD)
- `aci_bd_subnet` - Manage Subnets (infraSubnet)
- `aci_bd_subnet_is_subnet` - Bind Bridge Domains to L2 Out (infraBDGrp, infraBDGrpGrp)
- `aci_config_callback` - Provides callback and rollback provider functionality (configImport)
- `aci_config_snapshot` - Manage Config Snapshots (configSnapshot, configExport)
- `aci_contract` - Manage contract resources (infraContract)
- `aci_contract_subject` - Manage Intra-Contract Subjects (infraContractSubject)
- `aci_contract_subject_is_filter` - Bind Contract Subjects to Filters (infraContractSubject)
- `aci_domain` - Manage physical, virtual, bridged, routed or FC domain profiles (physDomain, vmmDomain, vmmDomainGrp, vmmDomainGrpGrp)
- `aci_domain_is_vmm_group` - Bind Domain to VMM Groups (infraDomainGrp)
- `aci_encap_pool` - Manage encaps pools (infraVlanEncapPool, infraVlanEncapPoolGrp, infraVlanEncapPoolGrpGrp)
- `aci_encap_pool_range` - Manage encaps ranges assigned to pools (infraVlanEncapPool, infraVlanEncapPoolGrp)
- `aci_epg` - Manage End Point Groups (EPG) objects (infraEPG)
- `aci_epg_monitoring_policy` - Manage monitoring policies (infraEPGGrp)
- `aci_epg_is_contract` - Bind EPGs to Contracts (infraContract, infraContractGrp)
- `aci_epg_is_domain` - Bind EPGs to Domains (infraDomainGrp)
- `aci_fabric_node` - Manage Fabric Node Members (fabricNodeMember)
- `aci_fabric_scheduler` - This module creates ACI schedulers

Fig. 7. Ansible module

Another important difference in Ansible with other tools, and the reason it has become so popular, is that it uses the simple YAML language, eliminating the need to learn an additional language.

Ansible is a Python-based configuration management and application deployment solution that is presently thriving in the automation management space. It brings together the benefits of numerous antiquated operation and maintenance tools. Ansible can essentially perform the same tasks as Saltstack and Puppet can.

Ansible is capable of managing, deploying, and configuring large numbers of hosts at once. To replace all hosts on a given Ansible control node, for instance, all that was required in the past was to switch one or more actions to each host.

Ansible operates through modules. It only offers a functioning structure. It is incapable of carrying out tasks. The activities are really carried out using Ansible modules. For instance, the service module is used to control service start, stop, restart, etc., while the copy module is used to transfer files to remote sites.

Agentless, or the absence of an agent, is one of Ansible's more distinguishing characteristics. It is more like a regular command than a piece of C/S software. Ansible only needs to be set up as a control node once on a host. Typically, the remote host is controlled using an ssh connection; Ansible or other additional services do not need to be installed on the remote system.

Users enter commands or playbooks at the server terminal, and the playbook is then broken down into plays in accordance with predefined rules, grouped into tasks that can be recognized by Ansible, called modules and plug-ins, and sent temporary files through SSH in accordance with the host list. Execute, then automatically remove the result after execution, and deliver it to the remote client.

The fact that the majority of Ansible's modules are idempotent is another distinguishing trait. Idempotency, as it is known, is the property that numerous actions or executions have the same effect on system resources. To stop the service, for instance, issue the `systemctl stop xxx` command. The consequence of successive stops will remain stopped since

nothing will be done if it discovers that the target service to be stopped is already in the stopped state. It has idempotence.

When executing a job, many Ansible modules first determine whether the target node wants to do so. As a result, you may confidently and safely let Ansible carry out the operation. The majority of the time, performing a task repeatedly will have no negative impacts. All operations under the system can be divided into two categories from the perspective of operation and maintenance operations.

Command execution: All terminal operations are composed of instructions for the system, and finally converted into electrical signals acceptable to the basic hardware to complete the task set. In terms of user behavior in O&M operations, operations other than file transfers can be called command execution.

The operations in Ansible can be divided into two categories from the perspective of operation and maintenance operations.

File transfer: File transfer activities include local and distant file transfers as well as modifications to the spatial and temporal forms of all files.

Execution of commands: All terminal operations consist of system instructions that are then transformed into electrical signals that the essential hardware can understand in order to finish the task at hand. Operations other than file transfers can be referred to as command execution in terms of user behavior in O&M operations.

Network administrators, system operators and maintainers, application operators and maintainers, desktop operators and maintainers, DevOps operators and maintainers, infrastructure operators and maintainers, and other multi-field operators and maintainers are all industries in which Ansible is applicable. It is also applicable to small-scale businesses that do not have operations and maintenance positions but require a certain amount of energy to maintain. Once they have a basic understanding, businesses and developers can start. It is also appropriate for medium-sized and big businesses that have the time, money, and manpower to invest in Ansible's further development and practicality.

CONCLUSION

In the global software development context, Scrum software development has proven to be quite effective[13]. It lessens some significant GSD difficulties, such as issues with coordination, control, and communication. According to the current trend, the application rate of scrum is gradually increasing over time. Scrum development ensures the speed and stability of delivery through an iterative and incremental development process. By strengthening teamwork and communication, the potential of members is maximized. It is able to adapt to fast changing requirements and guarantee good delivery quality. Through reasonable evaluation and refinement of user stories, combined with burndown charts, on-time delivery can be ensured. The future of agile development is inseparable from DevOps. The application of various DevOps

tools will become an important part of agile development to make up for the shortcomings of agile development in Operation.

REFERENCES

- [1] S. Chaouch, A. Mejri, and S. A. Ghannouchi, "A framework for risk management in Scrum development process," *Procedia Computer Science*, vol. 164, pp. 187-192, 2019.
- [2] E. Hossain, M. A. Babar, and H.-y. Paik, "Using scrum in global software development: a systematic literature review," in *2009 Fourth IEEE International Conference on Global Software Engineering*, 2009: Ieee, pp. 175-184.
- [3] K. Schwaber and M. Beedle, *Agile software development with scrum. Series in agile software development*. Prentice Hall Upper Saddle River, 2002.
- [4] L. Rising and N. S. Janoff, "The Scrum software development process for small teams," *IEEE software*, vol. 17, no. 4, pp. 26-32, 2000.
- [5] A. S. C. Marçal, B. C. C. de Freitas, F. S. F. Soares, M. E. S. Furtado, T. M. Maciel, and A. D. Belchior, "Blending Scrum practices and CMMI project management process areas," *Innovations in Systems and Software Engineering*, vol. 4, no. 1, pp. 17-29, 2008.
- [6] M. Blom, "Is Scrum and XP suitable for CSE Development?," *Procedia Computer Science*, vol. 1, no. 1, pp. 1511-1517, 2010.
- [7] H. Kniberg, *Scrum and XP from the Trenches*. Lulu.com, 2015.
- [8] M. Paasivaara, S. Durasiewicz, and C. Lassenius, "Using scrum in a globally distributed project: a case study," *Software Process: Improvement and Practice*, vol. 13, no. 6, pp. 527-544, 2008.
- [9] H. F. Cervone, "Understanding agile project management methods using Scrum," *OCLC Systems & Services: International digital library perspectives*, 2011.
- [10] P. Jalote, A. Palit, P. Kurien, and V. Peethamber, "Timeboxing: a process model for iterative software development," *Journal of Systems and Software*, vol. 70, no. 1-2, pp. 117-127, 2004.
- [11] C. Ebert, G. Gallardo, J. Hernantes, and N. Serrano, "DevOps," *Ieee Software*, vol. 33, no. 3, pp. 94-100, 2016.
- [12] S. Kumar, "Enforcing the Gnu GPL," *U. Ill. JL Tech. & Pol'y*, p. 1, 2006.
- [13] H. Holmström, B. Fitzgerald, P. J. Ågerfalk, and E. Ó. Conchúir, "Agile practices reduce distance in global software development," *Information systems management*, vol. 23, no. 3, pp. 7-18, 2006.

Assignment

GRADEMARK REPORT

FINAL GRADE

90/100

GENERAL COMMENTS

Instructor

Agile (40):36

Story Est (20): 18

SwDevOp Tool (40): 36

Feedback

=====

A good analysis of the Agile concepts but it could have been more clearly written in parts. Also you should have provided more examples of the process.

Story point analysis is very good, I would have liked an example to illustrate the process.

Nice discussion of Ansible, would have liked an example that more clearly demonstrated the benefits of this approach.

PAGE 1

PAGE 2

PAGE 3

PAGE 4

PAGE 5

PAGE 6

PAGE 7
