Bridging Agile Software Development and Human-Centered Design Principles to Advance Software Design Best Practices

RICHARD PAUL • MASTER'S OF DESIGN, DESIGN MANAGEMENT AND STRATEGY • SPRING 2020

KU THE UNIVERSITY OF KANSAS
Department of Design

Contents

Contents

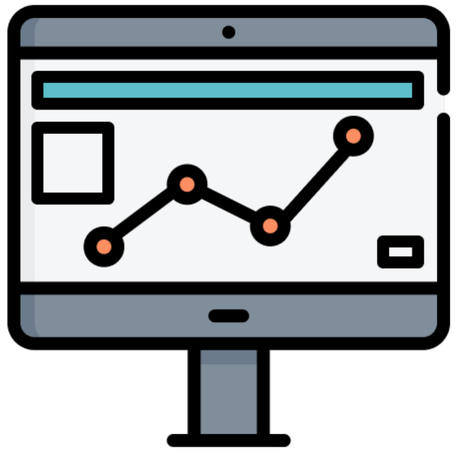
1: Abstract

Most software businesses today use the Agile development process to create their products. This process is engineering-centered and often disregards user research and design in favor of development speed and flexibility. In this thesis a bridging model will be proposed as a means to integrate and optimize the benefits of both Agile development and human-centered design (HCD) principles. Methods for integrating user research and design into the software development process are laid out. Requirements of the bridging model will be elaborated and evaluated.

2: Problem Statement

Software development lacks human-centered design thinking when defining features to build.

How can Agile development processes be improved by thoughtfully integrating established HCD principles and practices, thereby increasing validated, customer-focused value, desirability and market success of offerings?



3: Stated Biases

The author has over 20 years of design experience working in the software front-end development space. Therefore, expressed opinions and assumptions herein may be shaped on the basis of working relationships with many different software developers and engineers on myriad projects. The author has not, to date, served in executive positions nor been privy to decision-making that sets out goals and conditions of such projects.

4: Definition of Key Terms

- Software Development - The process a company uses to develop features of their products and services. This may encompass many disciplines including: Product Management, Research, Design, Engineering, Programming, QA and IT
- HCD - Human-Centered Design is a method of using empathy and research to identify user needs which inform the features of a product or service.
- UCD - User-Centered Design is similar to HCD.
- UX - User Experience is one of the design disciplines which uses HCD methods.
- IA - Information Architect specifies how sections of a software application are organized and named.
- Design Thinking - A collection of hands-on methods for implementing HCD.

4: Definition of Key Terms

- Waterfall Software Development - A method of breaking down project activities into linear, sequential phases. One phase cannot begin until the previous phase has finished.
- Agile Software Development - Formally created in 2001, this method for developing software focused on a process which was more collaborative, faster to market and more adept at changing direction in response to feedback.
- Scrum - A process framework of events and artifacts that emphasize teamwork, transparency, accountability and iterative progress. This is the framework most people understand as “Agile” since it is what most companies have implemented.
- Sprint - Mainly used in Scrum, this refers to a set period of time in which planned chunks of work are to be completed and made ready to review.
- QA - Quality Assurance is a systematic process for monitoring and evaluation of the quality of features of a product or service.

4: Definition of Key Terms

- IT - Information Technology is usually a section of a company who deal with creation and maintenance of systems and software which the company uses for software development and other portions of the business.
- Stakeholder - Usually refers to someone within the business who is either in charge of or makes decisions on the features and direction of software development projects.
- Developer - Business role usually focused on writing code for software products deployed on various platforms.
- Product Owner - A business role mainly responsible for managing the product backlog and making decisions on product direction and strategy.

5: Literature Review

“Design and programming
are human activities;
forget that and all is lost.”

— Bjarne Stroustrup (Creator of C++)

5.1: The Value of Agile

The trend toward using Agile software development seems to have accelerated around 2008 and in a 2017 survey of 601 software developers, 51% described their company as “Leaning toward Agile” while 16% were “Pure Agile” (Hewlett Packard, 2017). Even NASA in recent years has begun to challenge their internal development policies and turn toward a more Agile approach (Sublett, 2018).

Companies embracing Agile have seen huge benefits. In a global survey in 2018 of almost 1,300 IT companies, Harvard Business Review revealed that the companies identified as most agile had “60% greater revenue and profit growth than the rest” (Panditi, 2018). The top 18% of businesses measured on their scale were 4.1 times more likely to have a solid vision and strategy and 2.9 times more likely to “have teams skilled in the latest tools and trends.”

5.1: The Value of Agile

But, being an “Agile master” doesn’t always equate to being a market leader. Most surveys and reports lack information on the degree to which customer and user experience factors are leveraged. According to 13th annual State of Agile report, the top 20 reasons for employing Agile processes do not mention user needs or requirements. The top 3 reasons were: “accelerate software delivery”, “enhance ability to manage changing priorities” and “increase productivity.” Only 28% of companies reported using any kind of “Agile/Lean UX.” (CollabNet, May 2019)

Additionally, many organizations find it difficult to truly change their culture when adopting Agile. “The old, centralized, command-and-control system of management remains in place” (Kamer, 2017). Instead of self-organizing, autonomous teams, product owners mandate scope and deadlines. Plans and estimates need to be approved by a central committee. “Failure leads to blame, instead of learning and innovation” (Kamer, 2017).

5.2: The Value of Human-Centered Design

The intangible value of Human-Centered Design (HCD) is not as easily measured as other parts of a business. But, with low barriers-to-entry and easy access to scalable platforms, software companies are vulnerable to disruption more than ever. Customers won't hesitate to take their business elsewhere if a competitor offers a better overall experience. The importance of building long-term relationships with customers is dramatically underestimated by many top companies (Diller, Shedroff, Sauber, 2016).

For the most part, the value of human-centered design to business is undeniable. In a McKinsey report in 2018, the top-quartile of companies who they identified as being "strong at design" outperformed industry benchmark growth by as much as two to one (Benedict, Sheppard, et al. 2018). Microsoft's cultural transformation from "internally competitive" to a "growth mindset" with empathy has taken the company to new heights since the new CEO took over in 2014 (Waters, 2019).

There have also been criticisms of HCD and design thinking (Jen, 2018), which seem to center on misunderstandings or oversimplifications of design. However, there is a risk that a company's jump into HCD may be only skin deep and result in a kind of "innovation theater" which has the trappings of creativity while devaluing the strategic, purposeful, human-centered work (Kuznicki, 2018).

5.3: Conflict between Agile and HCD

When companies implement Agile, there tends to be a conflict between the Agile process and human-centered design. Usually the business is focused on development efficiency and accountability while design is focused on validating decisions and exploring options (Ng, Sep 3 2017, para. 6). Then, there are competing ideas between developers, designers and the business about the process itself.

For instance, one might receive very different responses when asking for the definition of when a task is “done” (Seiden, Mar 2019). The Scrum framework places a high value on being “done”. Points are estimated and assigned as a representation of the complexity of a particular task. Then, those points are awarded to the team when that task is completed. But, design work doesn’t fit in this same mold. Is a design task finished when the mockups are created? When approved by the stakeholder? Or is design work done when a feature is released and received well by customers?

5.3: Conflict between Agile and HCD

HCD also doesn't conform well to the inflexible 2-week sprints of the Scrum framework. They place an unrealistic expectation on designers to "create, test, refine, and deliver their output…with little of the context and big-picture thinking that suits consistent, user-centered designs" (Laubheimer, 2017, paras. 3-4). The team may develop "tunnel vision" where they lose sight of broad design and technology architecture decisions while focusing so much on individual feature tasks.

By placing too much emphasis on the backlog, organizations forget to do discovery up front. "Too often, teams look at the massive pile of work to be done and start imagining solutions in their head" (Toxboe, 2019, para. 14). The mountain of work that was already defined, estimated and shelved takes precedence over user feedback and research.

Some recommended improvements have proven beneficial such as doing quick user testing to evaluate designs instead of relying solely on stakeholder feedback (McInerney, 2017). Other suggestions such as skipping the step of assigning time estimates to tickets (Lazier, Skretch, 2018) have not worked well in this author's experience.

6: Analysis

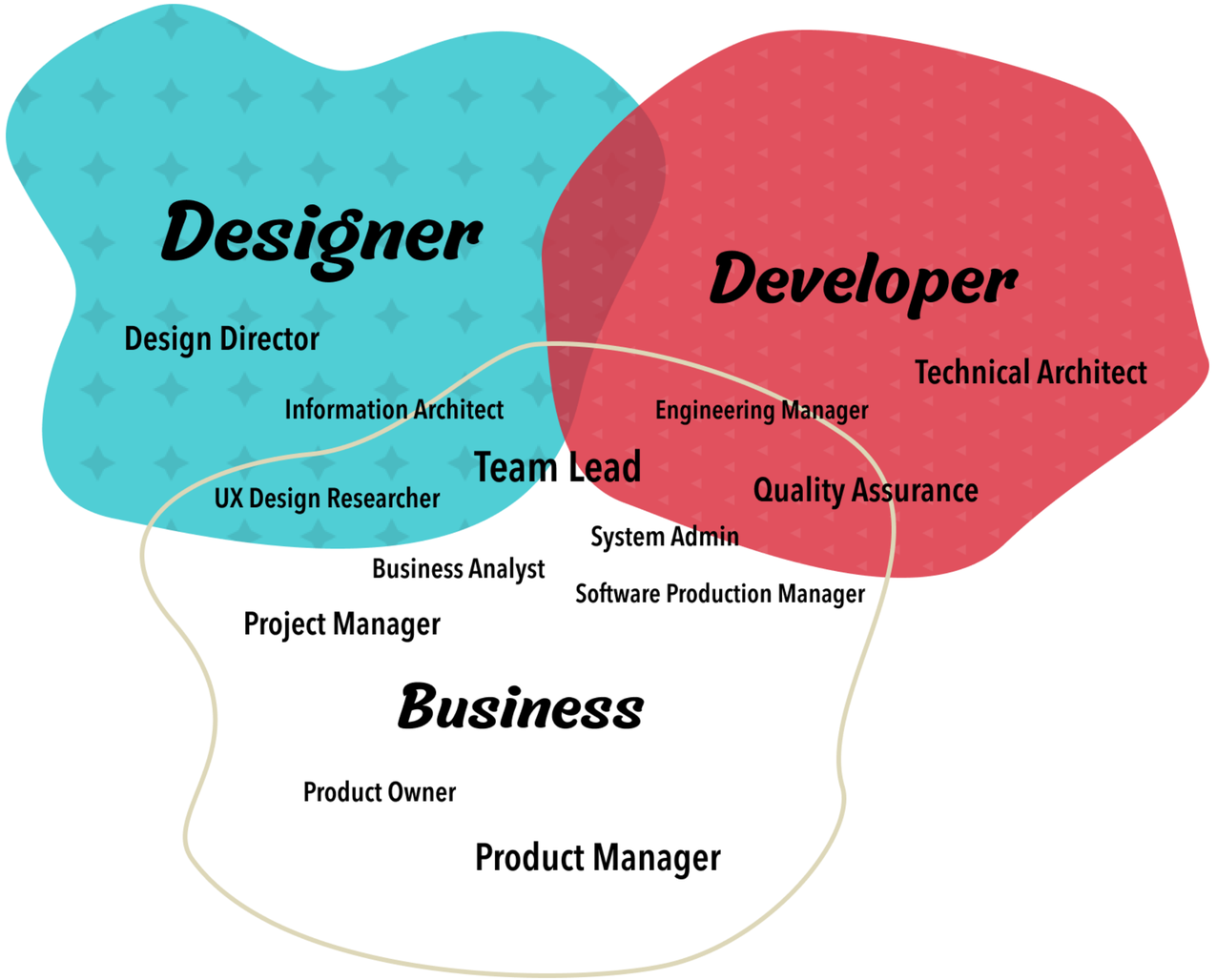
“Good design is a lot like clear thinking made visual.”

— Edward Tufte

6.1: Respondent Demographics

Of the 65 respondents surveyed, 25% classified themselves simply as designers with 18% as simply developers. 23% reported their titles as part of the business while 32% checked more than one box indicating they wear many hats such as: developer, team lead, technical architect and project manager.

The respondents also reported an average of over 20 years experience in the software development industry.

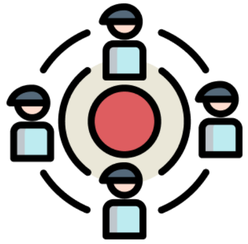


6.2: Where do you work?

64% reported they work in the same physical space as their co-workers while 38% indicated they work remotely with others in the same company.

Of those who work remotely, 52% said they work with colleagues in offices located in the US and around the globe while 28% are fully remote and work only from home offices.

In Person
64%



Remote
38%

52% Distributed offices

28% Work from home

Kansas
Massachusetts
New York
Washington
Texas

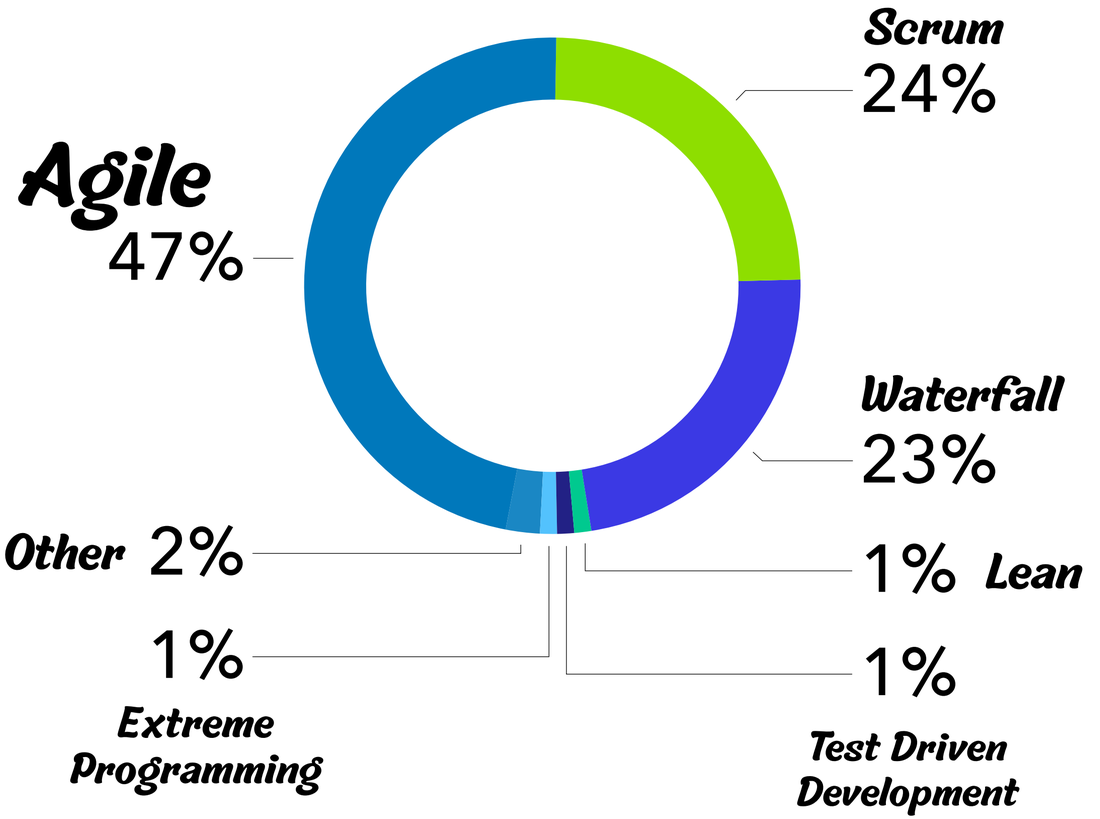
Boston
Georgia
Missouri
Tennessee

India
Singapore
Thailand
Canada
Netherlands
Australia
Argentina

6.3: Development Methodologies

47% of respondents described their development methodology as Agile while 24% specifically chose Scrum which is an Agile development framework with particular milestones and events.

23% reported they are still using Waterfall which is not an Agile framework, but a linear process where each part must be completed before the next can begin.



6.4: From whom do concepts for product features or specifications originate?

Origin of Feature Requirements

Business - 38

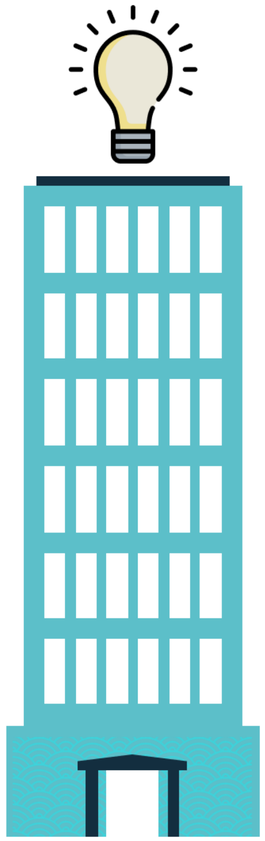
Executives and Project Managers 26
Market Research 5
Competetive Products 5
Marketing 2

User Research - 11

Developers - 5

Designers - 3

Customer Service - 3



Origin of Project Changes

33 - Business

16 Executives
12 Project Managers
4 Marketing
1 Other Business Stakeholders

6 - Customers

5 - Designers

2 - Developers

2 - QA

* Numbers represent the amount of respondents who chose a particular option.

In many companies, developers and designers working on software product features may not know the origin of the concepts or ideas behind the requirements they are given. When asked where they believe their requirements originate, respondents indicated the majority began with business executives or other stakeholders, though user research took second place.

When asked where they believe most changes originate during or after a project, the majority again indicated business executives and other stakeholders while only 6 indicated changes based on customer feedback.

This indicates a disproportionate amount of decisions about feature development come from the business instead of from customer needs and research.

6.5: What do you find most problematic?

When asked what part of development did they see as most problematic, the responses were grouped into the following categories using affinity mapping.

"We often don't build the right thing."

The responses showed a frustration with a lack of well-defined requirements and documentation before development begins. Communication and coordination between teams was the other biggest problem while several described the process as "pretending" to be Agile and not following its principles.

Pre-planning

★★★

- Lack of solid requirements or documentation before starting work.
- Features "lost in translation" from concept to requirements.
- Backlog not well-defined.
- No user research.

Other

- Open Source Software and licensing problems.
- Security
- Scaling popular development products.

Communication & Collaboration

★★

- Not enough coordination and communication between teams.
- Lack of collaboration.
- Changes not communicated.
- Too many changes to scope of project.
- Overlap of roles is not well defined or balanced.
- Managers who don't listen.
- Designers don't get dev constraints.

Working Process

★

- Pretending to be agile while still working waterfall.
- Lack of quick iteration and feedback.
- Bad estimates or project scope.
- Poor QA allocation and not enough testing.
- Technical limitations of old and/or overly complicated systems.
- Maintenance takes time away from new features.
- No cohesive process across all teams.
- Lack of documentation.
- Too many changes.

6.6: What would you change?

Respondents were asked if they could change one thing in their development process, what would it be? Responses were grouped into the following categories using affinity mapping.

"More planning so we can be less reactionary. "

The responses showed that most would prefer to have more input in the beginning of the process where they could influence the direction before development begins. Several indicated they would also like more time for user research and planning to clearly define requirements.

Pre-planning

★★★
- More input up front from:
  - Developers
  - Designers
  - Engineers

★★
- More time for the following before work begins:
  - User research
  - Prototyping
  - Planning
  - Documenting requirements
- Clearly define scope, goals and objectives.
- Less reactionary.

Communication & Collaboration

★
- More collaboration between teams.
- Faster feedback loops.
- Better stakeholder involvement and education on the process.
- Change is not bad as long as it's based on a real need.

Other

- More resources for development support.
- More project structure.
- Less process overhead.

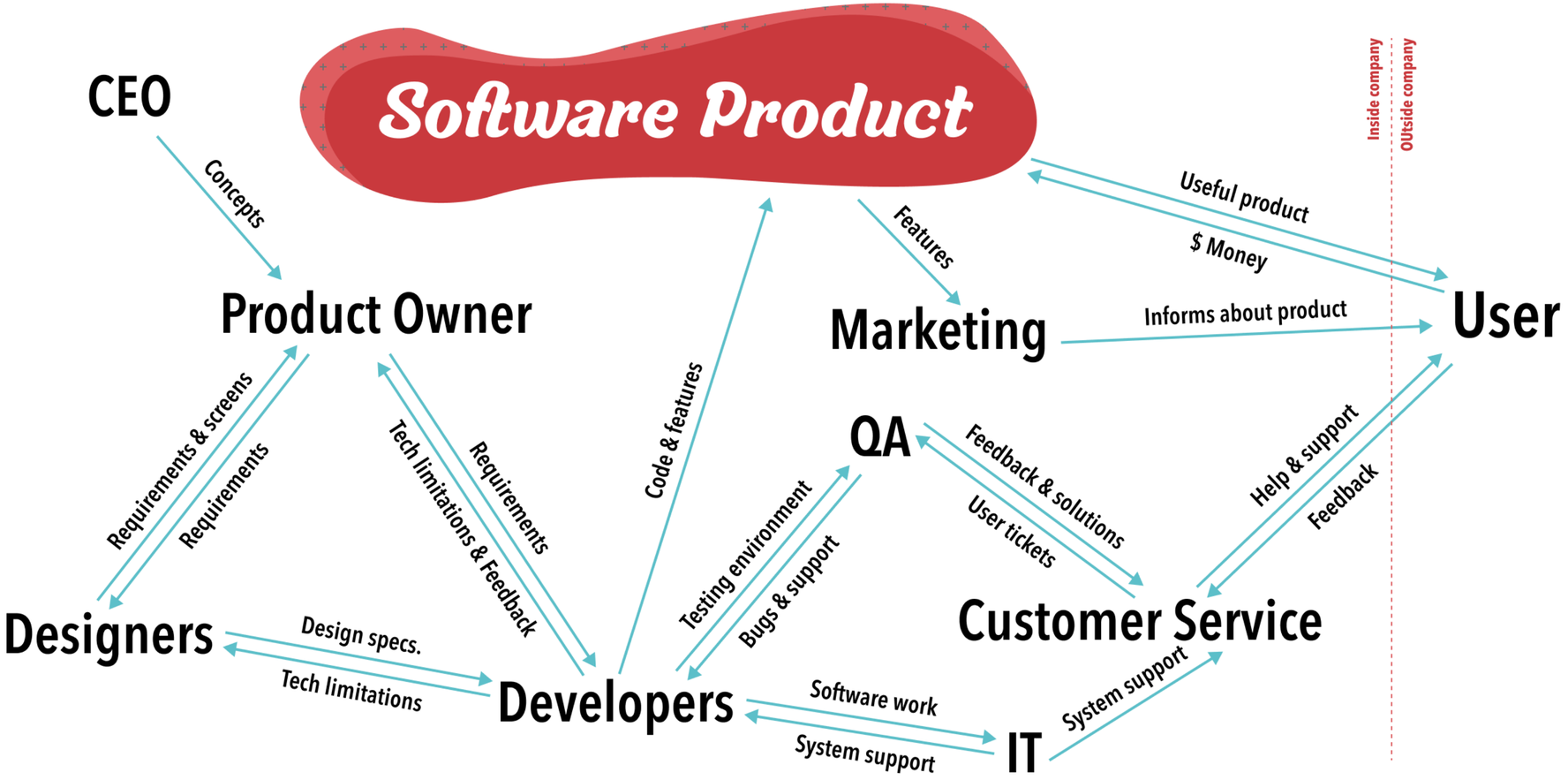
Working Process

- Follow a real agile process across all teams with management buy-in and understanding.
- Stable agile teams instead of shifting and swapping each project.
- Smaller, more nimble teams.
- Allow time for proper QA testing.
- Better backlog for future work optimization.
- Reduce minor task tracking to give the appearance of getting work done.

6.7: Descriptive Value Web

The book 101 Design Methods defines a Descriptive Value Web as "a way to visualize the existing set of relationships among stakeholders within a given context" (Kumar, 2012, pp. 150-151)

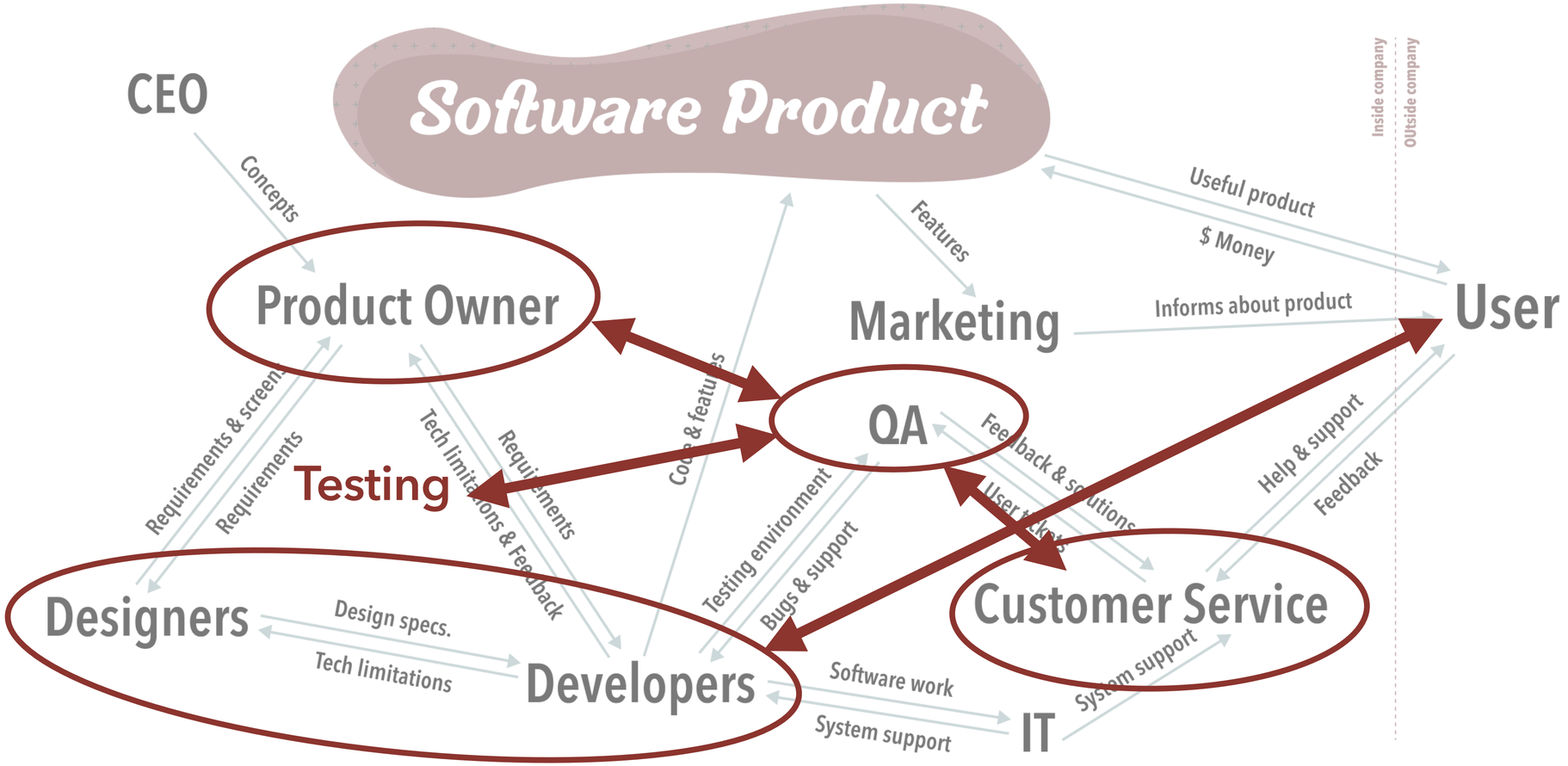
This web represents the typical relationships within a medium-to-large software development organization per the responses given during research.



6.7: Descriptive Value Web Gaps

The opportunities revealed include:

- More direct user involvement during the design and build process.
- More direct involvement of QA, customer service and executives for faster feedback on concepts with more up-front testing.



6.8: “Beth”: Java Developer

28 yo
Senior Developer
Married

“I think there’s a lot that gets ‘lost in translation’ from the top layer down to us. Project goals should be clearer and more fleshed out.”

Age: 28 years old

Occupation: Java Developer

City: Kansas City, MO

Personality: Introverted in meetings. Outspoken when tech decisions don’t make sense to her.

Her Story:

Beth graduated from KU with a Bachelor of Science in Computer Science. She found small jobs through a recruiter until she landed a position as a senior developer at a mid-sized company. She is recently married and they live in an apartment in an older part of town with their 2 dogs. She enjoys the technical aspects of her job and solving problems, but she doesn’t like the politics of corporate culture. As one of the few female developers, she feels pressure to perform at a high level and pushes to be involved in more decisions about the software development process. She would prefer a process that is stable and predictable. Too many changes make her feel like she is spinning her wheels and fear that the business will think she is doing a bad job.

Needs:

- Small chunks of features to estimate properly.
- A stable and predictable development process.
- Clear and refined requirements.

Motivations:

- Solving problems.
- Building products that appeal to the most customers.

Key Touch Points:

- Communication
- More predictable process
- Solve problems
- Incentive to produce code rather than outcomes

Frustrations:

- Changing requirements late in the development process.
- Visual designs which don’t account for technology limitations.
- Old, complicated systems which have not been updated.
- Lack of communication between teams and stakeholders.

6.8: “John”: User Experience Designer

34 yo
Senior UX Designer
Single

“It feels like we’ve just inherited decisions from previous groups. We’re just given solutions to be vetted rather than problems to be solved. ”

Age: 34 years old

Occupation: User Experience Designer

City: Lincoln, NE

City: Kansas City, MO

Personality: Outgoing and opinionated about design.

His Story:

John graduated from the University of Nebraska at Lincoln with a BFA, Graphic Design emphasis. John likes technology and likes to build his own web sites and projects in his off-work hours. After college he worked in the advertising industry and liked the fast pace, but he didn’t like the long nights and weekends. He decided to take a job as a user experience designer in a medium-sized company. He lives alone, but has an active social life. He is active in the local design community and has given presentations at several conferences and user groups around the country. He is very vocal about bad design and is sometimes perceived as a know-it-all.

Needs:

- Ability to collaborate with different teams.
- A problem definition, not a solution.
- Proper time and leeway to explore possibilities.

Motivations:

- Solving problems.
- Delighting users with a valuable product.
- Understanding what people need.
- Innovation and new technologies.

Key Touch Points:

- Collaboration
- Idea exploration
- Faster iterations and feedback
- Understand and influence business decisions.

Frustrations:

- Changing requirements without a user-specific reason.
- Not enough time to properly test.
- Arbitrary deadlines.

6.8: “Arjun”: Product Owner

56 yo
Product Owner
Married

“Many times, the feature that drives the bulk of the effort was something that ultimately wasn’t very important. We just didn’t have good conversations up front.”

Age: 56 years old

Occupation: Product Owner

City: Bangalore, India

City: Kansas City, MO

Personality: Methodical and soft spoken

His Story:

Arjun started his career as a developer in Bangalore. After rising up the ranks to manager, the company asked him to come to the US. He and his young family came over on an L1 visa. After 2 years, the company approached him to become a full time employee and helped him get his green card. He accepted and moved to the midwest. He works as a senior product owner in charge of one large segment of the business. He lives in a new development in a suburb of the city. His children are mostly grown with one in college. He cares about the company and the people he manages. He likes to solve business process problems.

Needs:

- A clear vision of where the company is going.
- Regular feedback from team leads about development momentum.

Motivations:

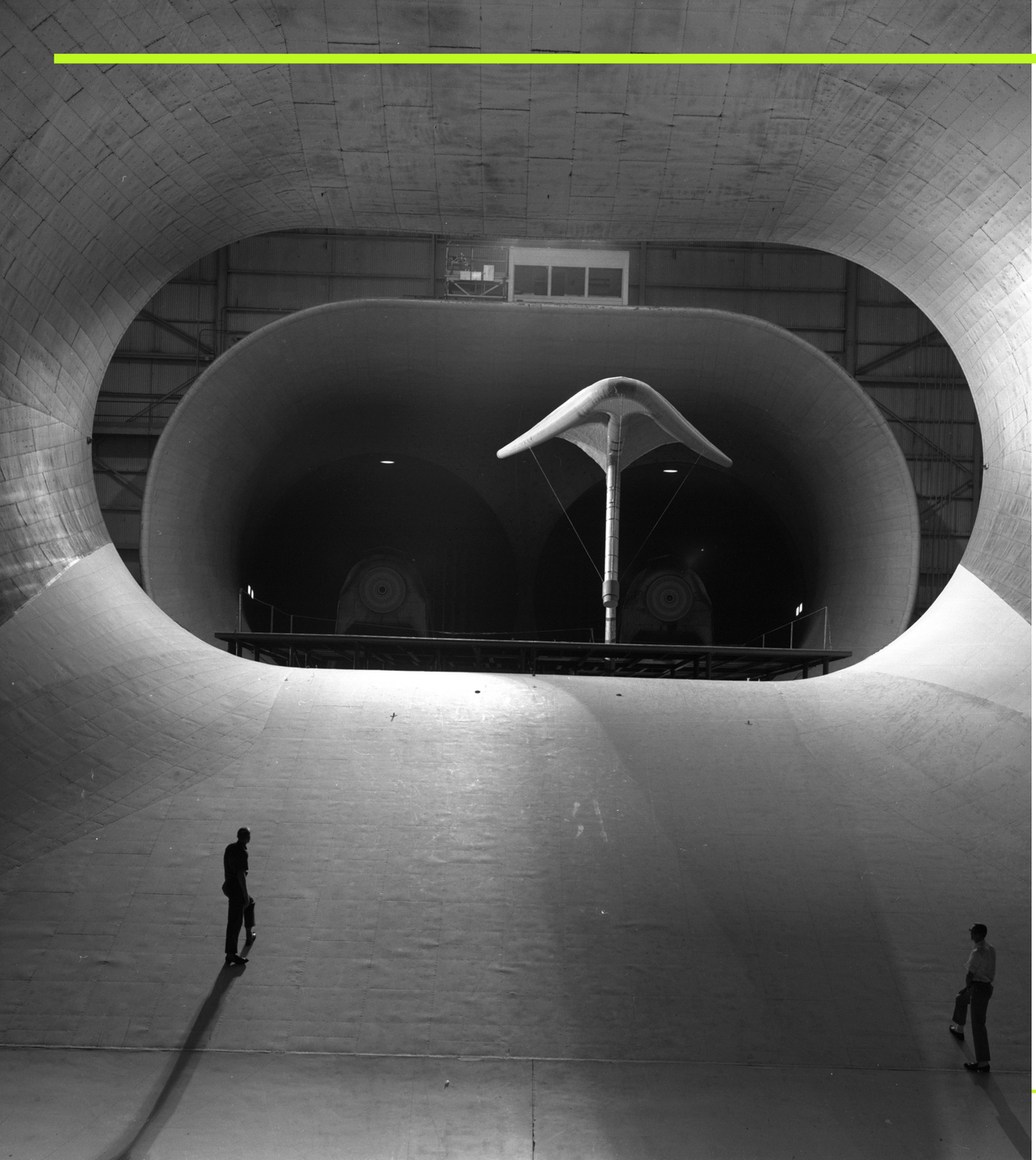
- Keeping the business growing
- Mentoring employees
- Refining process

Key Touch Points:

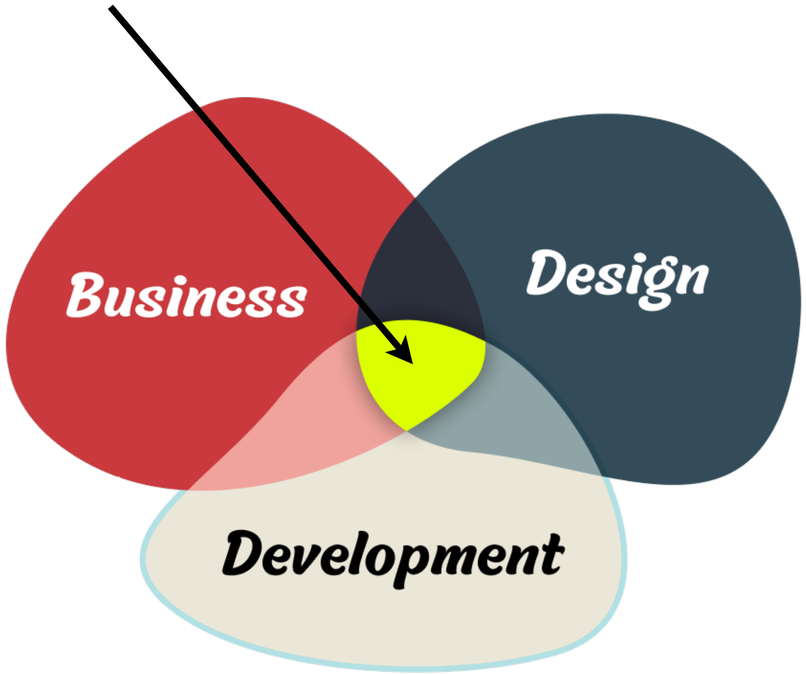
- Business value
- Speed to market
- Resource allocation
- Communication

Frustrations:

- Incorrect estimates
- Software bugs and inconsistencies
- Release delays
- Lack of communication between teams
- Features that don’t live up to vision



7: The Bridge Model



The Bridge Model brings together all departments of the software development lifecycle to better understand and integrate user needs and focus on outcomes rather than feature tasks during Agile software development processes.

7.1: Model Outline

Stakeholder Debrief

Document insights, outcomes, obstacles and measurements

Discovery

Visualization

Sprint Zero

Scrum

Delivery

Evaluate Success

Follow Up

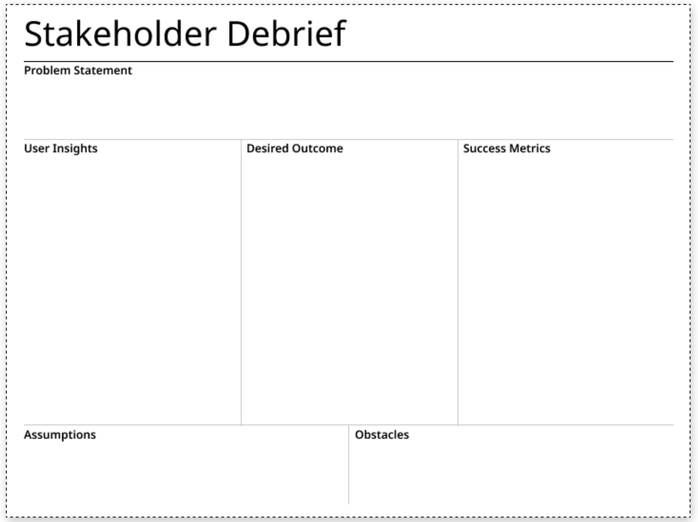
7.2: Stakeholder Debrief

To understand the reasoning and thought behind a product feature or specification, a stakeholder debrief should be performed. This can be held with the whole team or just one person to fill out the Debrief Form.

1. It is crucial to the success of the project to find the human insights behind a proposed feature. Insights should be crystalized into a solid problem statement with the help of the stakeholder.

2. The desired outcome is secondary only to user insights. This should be framed as the benefit to the user, although business outcomes could be captured as well.

3. Finally, we must understand how we will know if the project is a success. Tracking these metrics should begin immediately if possible. This will provide a baseline against which success may be compared after feature release.



Who:

- Stakeholder
- Design Lead

Optional:

- Team

Although less crucial, assumptions and obstacles could also be captured which will clarify the stakeholder's underlying mental model for this project.

7.3: Document Insights

The underlying insights, outcomes, metrics, assumptions and obstacles revealed by the stakeholder debrief should be documented in a central location accessible by the entire team. User insights should be carefully annotated to allow for reference throughout the rest of the project.

The status of the documentation should be communicated to all involved once complete.

Who:

- Design Lead

7.4: Discovery

With the insights documented, the team should be brought together to discuss solutions to the problem statement. This portion of the process should deliver the following:

- **Loose requirements** based on the desired user outcome.
- **Low-fidelity sketches** and workflow diagrams of possible processes.
- **List of assumptions** from the team about parameters, constraints and technology.

Who:

- Entire Team

"Don't stop asking 'what customer problem is this solving?' until the team truly understands the problem. That's how we deliver the best products and services. "

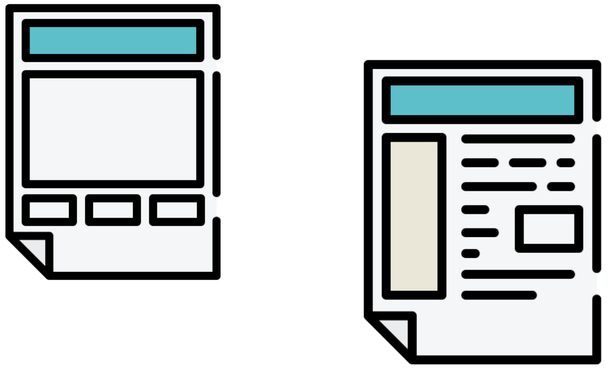
– Jared Spool

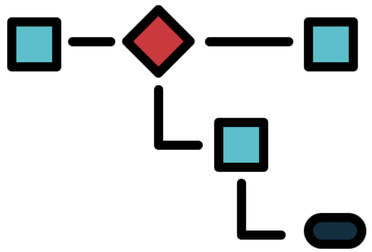
7.5: Visualization & Validation

Designers should collaborate with the team to visualize and clarify the requirements created during the Discovery phase. The goal is to reduce ambiguity. Make sure everyone on the team is talking about the same things when discussing the project.

Collaboration should also involve users outside the business. Testing with real users can quickly reveal incorrect assumptions about the functionality of a feature before spending development time.

This phase should produce wireframes, interactive prototypes and user flow diagrams.





Who:

- Design Lead
- Team Leads
- Team Members
- Stakeholders
- Users

7.6: Sprint Zero

Once the solution has been verified with end users and insights have been re-incorporated into requirements, the team is ready to begin work. The first sprint should be used by each part of the team as a "mini-discovery" of technical requirements.

- **Developers** should assess the current state of the code and technology.
- **Designers** should assess graphical and other UI elements.
- **IT** should assess the systems needed to produce data consumed by the software.
- **QA** should assess testing schedules and scope.
- **Customer Service** should assess new features and support materials needed.

Methods of measuring the success of new software features should also be evaluated and planned.

Who:

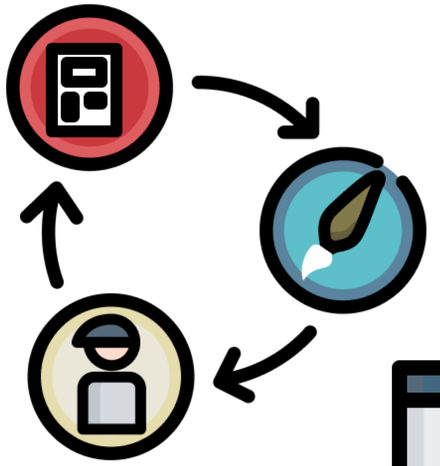
- Entire Team
- Outside Teams

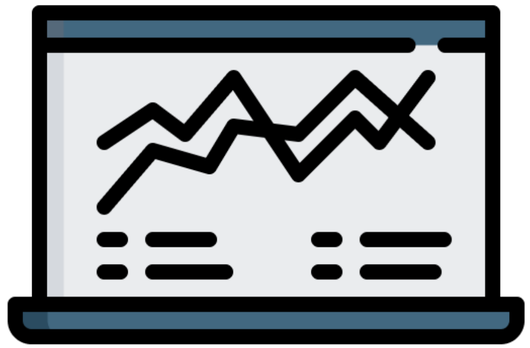
Once these assessments have been performed, the team should re-assemble to create chunks of work with broad estimates for their completion during the entire project.

7.7: Scrum & Agile Development

With work broken out into chunks, the regular Scrum process can begin. Design and development teams should produce working software each sprint.

Most importantly, users must be involved to evaluate the project direction as software is developed. Dogged adherence to backlog tasks without evaluation must be avoided at all costs. Team members and stakeholders should embrace changes driven by user feedback and research.





Who:

- Entire team

7.8: Delivery

When the software is complete, tested and verified by users, it should be released. The team should coordinate with the following groups to make sure the release goes smoothly:

- **QA** - for testing in the production environment.
- **Customer Service** - for support and direction to users experiencing problems.
- **Marketing** - for advertising and manuals.
- **IT** - to coordinate backend systems and data.

Who:

- Entire team

7.9: Evaluate & Follow Up

Once the software is delivered, the success metrics should be evaluated to know the results of the software project.

Feedback received during and after release should be captured as future enhancements. But, these items should not be maintained in the backlog unless work will start immediately.

When future improvements are prioritized, these recommendations should be re-evaluated and new outcomes should be created rather than assuming relevance.

Who:

- Team Leads
- Stakeholder
- Customer Service

8: Conclusion

Improving user-centered planning before an Agile software development project begins will produce outcomes that better fulfill user needs. The Bridge Model accomplishes this by:

- Understanding and basing outcomes on user needs.
- Listing biases and assumptions.
- Collaborating and communicating more often and coordinating with other teams.
- Involving end users in the process from the beginning and throughout the development process.

9: References

Chandler, C., et al. (Dec 22, 2017). Fortune's 2018 Business by Design List. Fortune. Retrieved from http://fortune.com/2017/12/22/business-design-apple-airbnb-tesla/

Waters, R. (Dec 21, 2019). Satya Nadella brought Microsoft back from the brink of irrelevance. Retrieved from https://www.latimes.com/business/technology/story/2019-12-21/satya-nadella-reinvigorated-microsoft

Panditi, S. (Mar 22, 2018). Survey Data Shows That Many Companies Are Still Not Truly Agile. Harvard Business Review. Retrieved from http://hbr.org/sponsored/2018/03/survey-data-shows-that-many-companies-are-still-not-truly-agile

Hewlett Packard. (Mar, 2017). Agile is the new normal. Hewlett Packard Enterprise. Retrieved from https://softwaretestinggenius.com/docs/4aa5-7619.pdf

Jeremiah, J. (Mar, 2017). Survey: Is agile the new norm? Tech Beacon. Retrieved from https://techbeacon.com/app-dev-testing/survey-agile-new-norm

Varhol, P. (Aug 26, 2015). To agility and beyond: The history—and legacy—of agile development. Tech Beacon. Retrieved from https://techbeacon.com/app-dev-testing/agility-beyond-history-legacy-agile-development

9: References

Nyce, C. (Dec 8, 2017). The Winter Getaway That Turned the Software World Upside Down. TheAtlantic.com. Retrieved from https://www.theatlantic.com/technology/archive/2017/12/agile-manifesto-a-history/547715

Ng, S. (Sep 3, 2017). UX designer unhappy in Agile? It might be because of these 4 reasons. UX Collective. Retrieved from https://uxdesign.cc/ux-unhappy-in-agile-it-might-be-because-of-these-4-reasons-921790326cf

Seiden, J. (Mar 21, 2019). Agile Won't Get You to Done: Here Are Four Ways to Fix That. UX Collective. Retrieved from https://uxdesign.cc/agile-wont-get-you-to-done-here-are-4-ways-to-fix-that-4e6b83b85a4a

Laubheimer, P. (Sep 24, 2017). Agile Is Not Easy for UX: (How to) Deal with It. Retrieved from https://www.nngroup.com/articles/agile-not-easy-ux/

Toxboe, A. (Apr 17, 2019). Don't trust agile alone to build successful products. Retrieved from https://ui-patterns.com/blog/dont-trust-agile-alone-to-build-successful-products

Hattangadi, V. (Dec 4, 2017). Design thinking: How new buzzword has taken India Inc by storm. Financial Express. Retrieved from https://www.financialexpress.com/opinion/design-thinking-how-new-buzzword-has-taken-india-inc-by-storm/959170/

9: References

Linders, B. (Apr 19, 2018). 12th State of Agile Report Published. InfoQ.com. Retrieved from https://www.infoq.com/news/2018/04/state-of-agile-published/

Sublett, H. (Aug 8, 2018). UX in Space. SolutionsIQ. Retrieved from http://www.solutionsiq.com/resource/agile-amped-podcast/ux-in-space/

Diller, S. & Shedroff, N. & Sauber, S. (2016). Blind Spot: Illuminating the Hidden Value of Business. Brooklyn, NY: Two Waves Books

Gothelf, J. & Seiden, J. (2016). Lean UX: Designing Great Products with Agile Teams. Sebastopol, CA: O'Reilly Media.

McInerney, P. (May 15, 2017) UX & Agile: Dream Team or Odd Couple? Medium.com Retrieved from http://medium.com/design-ibm/https-medium-com-paulmci-ux-agile-dream-team-or-odd-couple-1e06f9a01a26.

Siroky, D. (Sep 2018) Agile and DevOps are failing in Fortune 500 companies. It should be a wake-up call to all of us. Plutora.com. Retrieved from https://www.plutora.com/blog/agile-devops-failing-fortune-500-companies-wake-call-us

Lazier, M. & Skretch, J. (Jul 17, 2018) Experiments in Tweaking Agile for UX. DigitalGov. Retrieved from http://digital.gov/2018/07/17/experiments-in-tweaking-agile-for-ux/

9: References

Jen, N. (Aug 2017) Design Thinking Is Bullsh*t. Adobe. Retrieved from http://99u.adobe.com/videos/55967/natasha-jen-design-thinking-is-bullshit

Kuznicki, M. (Apr 13, 2018) 3 Signals of “Innovation Theatre” to Watch Out For. Medium.com. Retrieved from https://medium.com/the-moment-is/3-signals-of-innovation-theatre-to-watch-out-for-8c2e29d4dcf3

Blomkvist, S. (2005). Towards a model for bridging agile development and user-centered design. In Human-centered software engineering—integrating usability in the software development lifecycle (pp. 219-244). Springer, Dordrecht.

Glomann, L. (2017, July). Introducing ‘human-centered agile workflow’(HCAW)–an agile conception and development process model. In International Conference on Applied Human Factors and Ergonomics (pp. 646-655). Springer, Cham.