How we work

Digital Natives working methods

22 October 2014

How we work

Digital Natives (DiNa)'s business goal is to take part in and support the entire development process and life-cycle through mutual communication, trust and partnership.

In this document we describe the complete lifecycle of a product from the beginning of development to live, continuous releases.

We help our clients turn their ideas into products by understanding their business goals and providing the right methodology and technology to achieve those goals.

Our clients have a vision, and often market knowledge, and we have the digital product development knowhow and experience to turn visions into realities.

Business model of Digital Natives

DiNa's approaches product development as a service (rather than a precisely pre-defined project), whichallows DiNa and its Clients to develop innovative and outstanding products together. This service gives access to our knowledge, capacity and superior technical abilities.

DiNa provides its service by followinglean product development and agile software development methodologies, allowing the Client to bring a validated, working version of the product to the market. We provide this service in a team-renting model with monthly reonciling, which means we are committed to continuous, long-term product development.

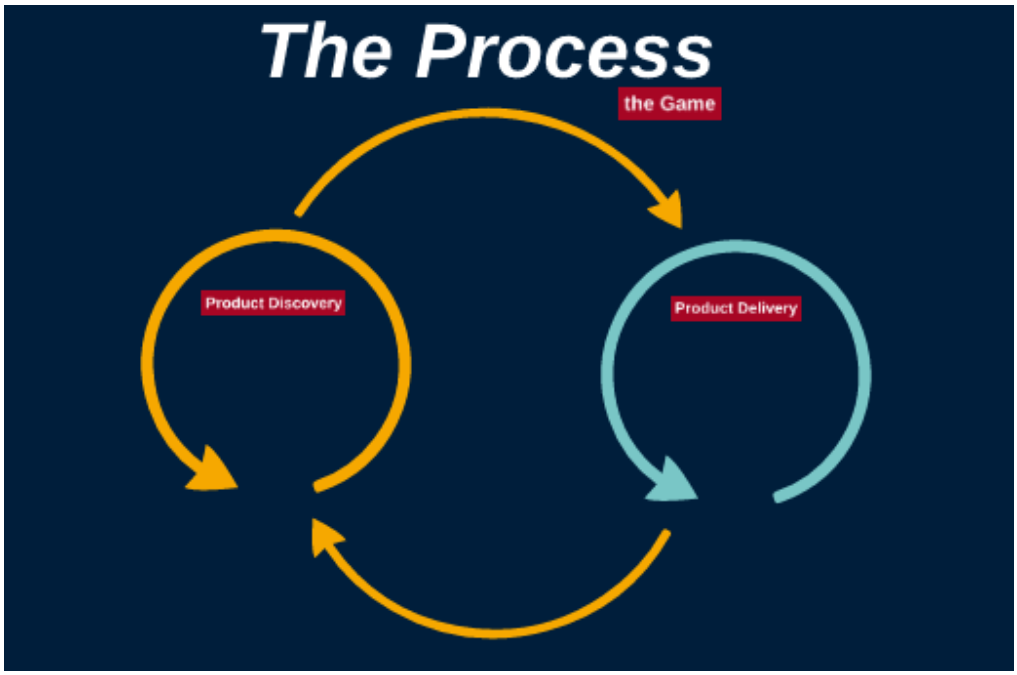
Based on our 10 years of industry experience, we do not believe in a "project mindset" based development - rather,we are committed to development that is driven by creative, market needs instead.

We believe that succesful product introduction relies on a deep understanding of users' needs, validated hypotheses, testing and learning through quick and cost efficient iterations.

Product development cycle

At DiNa, product development consists of two major cycles: **discovery** and **delivery**.

In the following sections we will walk you through each phase in further detail so you have a clear understanding of the entire process and the tasks associated with each step.



Source: Gábor Kardos of Prezi

Team roles

The team is dedicated to the product, therefore they feel responsible for its quality. The typical roles in a team are separated into Discovery and Delivery roles, although there are overlapping roles as well.

The **Discovery team** is responsible for evaluating user behavior, mapping out the possible directions for product development , validating them and prioritizing themfor development.

The **Delivery team** is responsible for developing shippable software based on the Discovery team's preparation work.

Discovery

Goal

Validate business problem / solutions and translate the idea/concept into a comprehensive product roadmap that is the basis for the development work.

Client's benefit

Everything can be coded, the question is, what isworth coding? We not only help to design and code your solution, but we develop only those features that users really need and will love to use. Upon completion of this initial conceptual planning, clients will have a set of documents that provide a carefully thought-out, high-level overview of a product development roadmap for 3-12 months.

DiNa's role

Validate possible development plans by user interviews, rapid prototyping, and researching technical feasibility (pro / cons).

DiNa supports clients throughout this process, providing lean product management, design and technical advice (and emotional support as needed!) based on years of experience. Clients may have some work already prepared but starting from scratch is also feasible from our end. We are flexible and happy to get involved at any stage.

The Discovery phase begins after thecontract has been signed.

Key tasks and outputs

- **Lean and Business model canvas** - a detailed overview of the product idea which describes who will use it, for what, what problem it solves and suggests possible business models. This document will be continually revised and expanded during the entire Conceptual Planning cycle.
- **Problem / solution validation** - by user interviews. We help prepare interview questions, schedule key users, and analyzeand create reports from the interview sessions.
- **Prototyping** - solution validation by user testing. We create paper or clickable prototypes to test web pages / mobile views and workflows. These are designed to better understand whether proposed workflows and layouts are understandable for the avarage user.
- **User workflows** - a breakdown of all major user activities by roles. We like to use UML diagrams.
- **Product roadmap** - a timeline of feature releases and scope for each development milestone. This document describes the desired velocity of development and which features are included in each phase, eg: Prototype, Alpha release, Beta release, MVP, etc.
- **Product backlog** - a detailed catalogue of features described as user stories. This document sets the priorities and acceptance criteria of each milestone in the development cycle. In other words, this is a more detailed view of the roadmap for a shorter time period.
- **Rough scope estimation** - based on the product backlog, we prepare this document to help clients align the scope of the product with their available budget. More detailed and accurate estimations are developed during Sprint cycles (see below).

Discovery team roles

- **Product Owner (PO)** - assigned by the client, responsible for making decisions, prioritizing tasks and accepting delivery.
- **Quality Assurance? or Business Analysts (QA or BA)** - help translate the business needs to technical requirements by analog / user approach and help to define acceptance criteria.
- **Senior Developer (Sendev)** - supervises the technical aspects of the project, selects the appropriate technologies, tools and platforms.
- **Lean Consultant (LC) -** helps to keep focus on the “validated learning method.”
- **UX / UI designer, interaction designer (UX/UI)** - in charge of the software workflows from the user perspectiveand helps put together prototype.
- **Frontend Developer (FD)** - helps put together interactive prototypes if needed.
- **User Tester (UT) -** organizes and leads user testing sessions; validates and summarizes test results, generates reports from the prototype testing.

One person can have more than one role.

Delivery

Goal

Develop a production-ready release of the product within time and budget estimates, based on the discovery team's work.

Client's benefit

Only the highest prioritized features are developed, saving precious resources and allowing for learning and pivoting during development.

DiNa's role

Team formation and responsibility for all software development. The development cycle starts with a kick-off meeting where the whole team meets for the first time.

Development starts after the discovery team has delivered required outputs.

Estimated timeframe

When starting with an idea from scratch, the time from concept to MVP (Minimum Viable Product) is typically between 8-12 weeks. Later down the line we can do shorter iterations and live releases to production every day, as needed.

"A Minimum Viable Product is the smallest thing you can build that delivers customer value (and as a bonus captures some of that value back)."

- Ash Maurya

Iterative development

Different product life-cycle phases requires different approaches to the delivery. At DiNa we prefer to use Scrum and Kanban- style development.

If the business requires lots of new features, Scrum can offer more benefits with stricter timeframes and “frozen” short iteration handling. Once the product is out and live and there are many ongoing support needs that can not be delayed for the next iteration, Kanban might be a better solution.

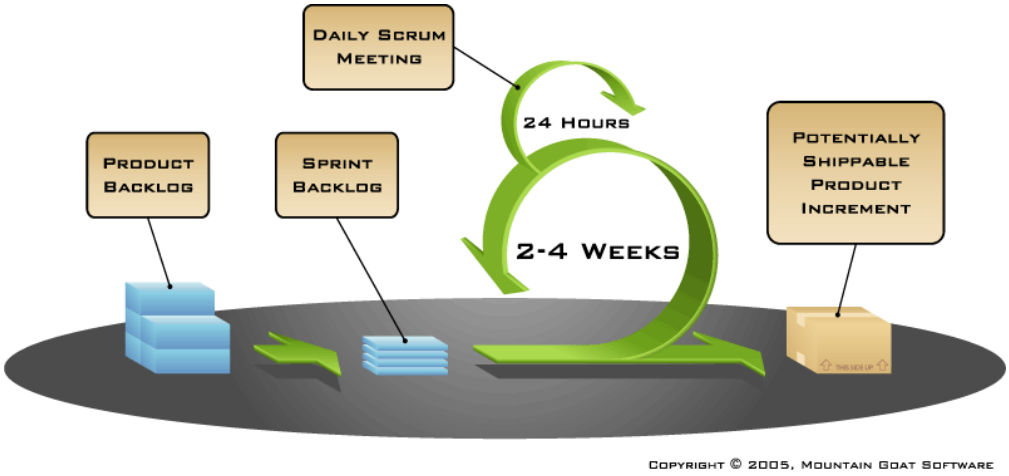
These methods can be mixed and the team can switch between the methods according to the current product development expectations.

Scrum

Development occurs in 3-stage loops called iterations, or “sprints.” These usually occur weekly or biweekly, depending on the workload and available resources.

The first stage of a sprint is planning (usually 2-3 hours for a 1 week long sprint), followed by development (5 work days) and then review (1-2 hours). Once a product is in live usage, there will be continuous support needs; these tasks are handled more flexibly and the team is able to insert the urgent fixes into the daily work.

Find more about the Scrum rules at Scrum Org’s official website.

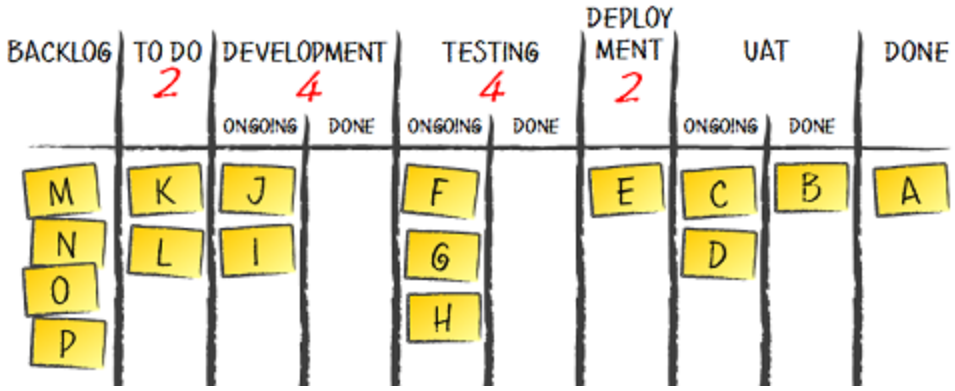


Kanban

Kanban is a method for managing knowledge-based work with an emphasis on just-in-time delivery without overloading the team members. In this approach, the process, from definition of a task to its delivery to the customer, is displayed for participants to see. Team members pull work from a prioritized queue.

In the context of software development, Kanban is simply a visual process management system that displays what to produce, when to produce it, and how much to produce, inspired by the Toyota's now famous lean manufacturing and production system .

Find more about Kanban rules at Wiretech website.



brodzinski.com

Delivery team roles

- **Scrum Master (SM)** - responsible for enforcing scrum principles and removing any impediments from the team's work (this is associated with Project Manager role).
- **Senior developer (SDev)** - assures code quality, merges developers code, manages updates and deployments.
- **Developers (DEV)** - web backend & frontend and mobile developers are the core team members, responsible for writing code and conducting automated tests.
- **Tester (QA)** - conducts manual testing of the software based on the user stories' acceptance criteria and validating the expected user experience.

QA process

Acceptance criterion definition

Based on the Product Owner's business needs, the QA helps to define the acceptance criterion and finalize it with the Product Owner.

Automated testing

Based on the acceptance criteria, developers create automated tests. First are unit tests that areaable to test logically the code, eg. 4+4 = 8. Secondly developers create integration tests that test complete, pre-defined workflows with a pre-defined database in different browsers, eg. in Google Chrome 1) login to the site as an admin, 2) go to profile, 3) change privacy settings. Running automated test cases like these on the entire software every time new code is committed to the code base helps ensure that a new feature or fix doesn't break the software.

Manual testing

Once the developers finished a task AND all the pre-written automated tests are green, it means a software componentis ready for manual testing. The QA tests it manually based on the acceptance criterion and "human sensibility." If something comes up, the QA pushes it back to developers; otherwise if it works properly, then it is ready for acceptance of the Product Owner or similar dedicated person.

Releasing

Once the Product Owner finds everything working properly, the software component can be released to the production environment. If the Product Owner finds "something went wrong" during the testing, the software will be pushed back to developers for further fixing.

Bugs - why are there bugs?

Bugless software is an illusion, although our goal is to create software as bugfree as possible.
We carefully design software and test is using a well thought QA process during development, but there are some scenarios when bugs still show up in the product.

The explanation of why-are-there-still-bugs-in-the-software is that there are some user scenarios that are not predictable (data sensitive bugs, edge use cases, etc.), and yes, we are human beings and we make mistakes sometimes.

Although we see bugfixing as normal accompaniment todigital product development, we do measure bug fixing ratio of our work as a KPI and analyze the causes of the bugs.We use this information to continuously improve the QA and design process.

Bugs Definition

As the definition of a ‘bug’ can vary, for the purposes of this document we consider a ‘software bug’ to be any error, flaw, mistake, or failure that produces incorrect or unexpected results, or a failure that causes the product to behave in unintended ways. However, bugs are not equals to lack of proper design - for instance a business logic / user interface thatwas planned and implemented in a way which doesn't cover the exact needs is not a bug, but feedback for the discovery team.

Software Product bug categories by wikipedia:

- **“A”,** Critical bugs. There is no way a customer can accomplish a given task, therefore the user is blocked and not able to access major feature(s) of the software.
- **“B”** There is workaround in the system or the bug does not block the user from accessing the major features of the software.
- **“C”** "UI" or "visual defect." For example, a missing image or displaced button or form element.

Exceptions

There are some cases when a bug shows up, but because of lack of proper design or execution, the warranty does not enter into force. These cases are:

- If the Principal (Product Owner) fails to hand over the project, namely: not doing manual testing after the sprints and/or at the release sprints (not later than 1 workday calculated from sprint handover).
- If the Principal directly gives order to Contractor to not implement sufficient automated or manual tests into the software.
- If the Principal directly gives order to Contractor to not follow QA processes (planning, manual testing).
- If the malfunction's cause is a 3rd party API, or a hosting environment change in the background.
- If the Staging or Production software environment setup does not meet with requirements.
- If 3rd party programmers modify the code base.