

Scrum

The Product Owner – It Is All About ROI

Dr. Ronen Bar-Nahor

Changing Mind Set

“... but it is not just a development gig..”

- It's mainly breaking paradigms and changing the organizational mind-set
 - Development can't estimate & commit on what they do not fully understand
 - Business can't freeze scope for a long time and can't sign-off on specs they do not fully understand.
 - As users see the software, they come up with new ideas
 - **If we can't perfectly predict a schedule, we can't perfectly say what will be delivered.**
 - **Development of software is an empirical process**

Product Owner

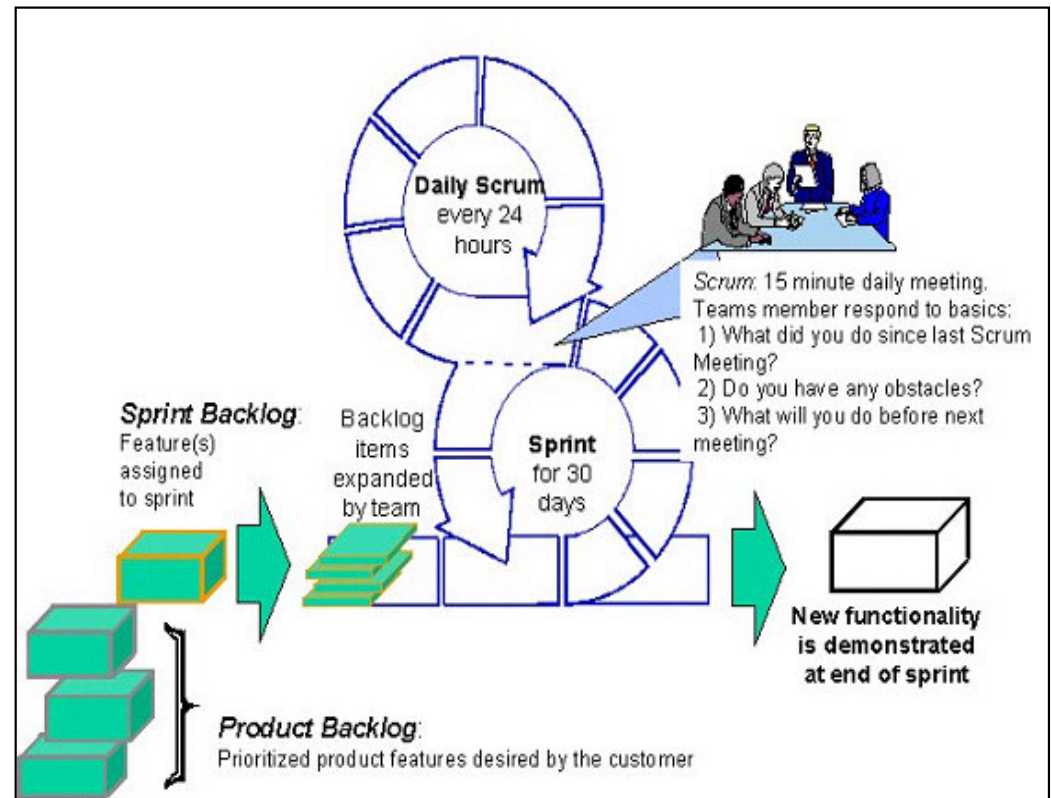


- Is responsible for the profitability of the product (ROI)
- Synthesizes interests of stakeholders
- Lead the “customer team”
- Break down stories into right sizing (**Just-In-Time and in Just-Enough Details**)
- Constantly Prioritize the backlog
- Negotiates with the Scrum team:
 - Sprint Goals , Sprint Backlog Items, Done Criteria
 - Final arbiter of requirements questions
 - Accepts or rejects each product increment

This requires high bandwidth for in-bound and out-bound communication

Product Owner Role In The Process

1. Stories & Backlog Management
2. Sprint Planning
3. Sprint Execution
4. Daily Scrum
5. Sprint Review
6. Sprint Retrospective



What problem do stories address?

- Software requirements is a communication problem
- Those who want the software must communicate with those who will build it

Value Driven User Stories

- **“Stories are a promise for a future conversation”**, a reminder that describe the business functionality/Value we need to deliver [Ron Jeffries]
- A story is a link between feature, user/role and value
- May assume template as:
 - – **“As a <role> I can <function/goals> so that <rationale/benefits>.”**
- **Where do we document the requirements?**

Ron Jeffries' Three Cs

Card

- Stories are traditionally written on note cards
- Cards may be annotated with estimates, notes, etc

Conversation

- Details behind the story come out during conversations with product owner

Confirmation (DoD)

- Acceptance test and “rainy cases” confirm the story was coded correctly

Conversation

- **As a frequent flyer, I can cancel a reservation.**
 - Does the user get a full or partial refund?
 - Is the refund to her credit card or is it site credit?
 - How far ahead must the reservation be cancelled?
 - Is that the same for all hotels?
 - For all site visitors? Can frequent travelers cancel later?
 - Is a confirmation provided to the user?
 - How?

So where are the details?

Is the refund to her credit card or is it site credit?

- Details as Definition of Done (DoD) → essentially tests
 - Details as sub-stories
-
- Augment them with written documentation as appropriate
 - Business rules
 - Data dictionaries
 - Use cases
 - Examples of inputs and expected result

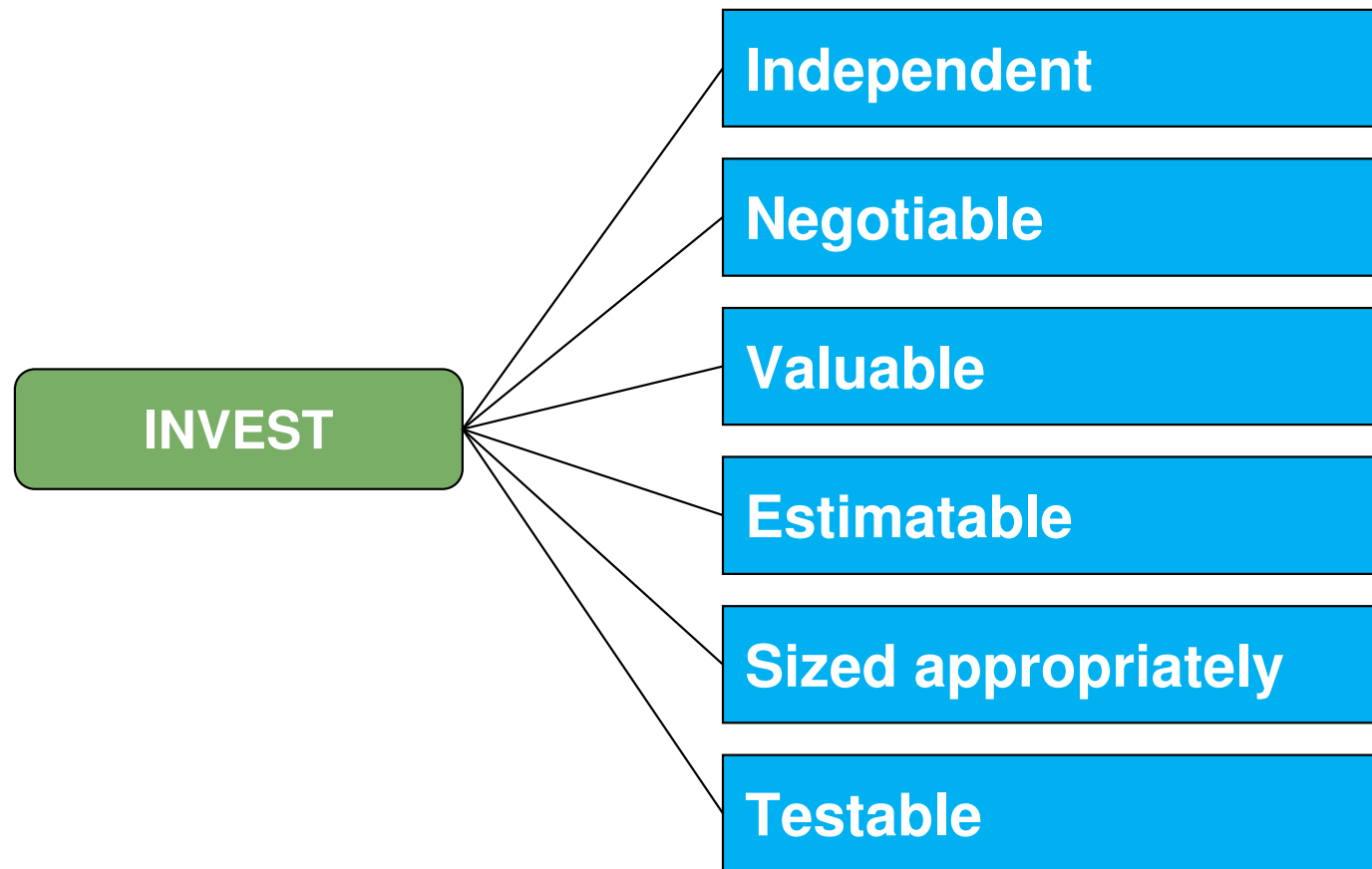
Confirmation

Definition of Done (DoD)

- Condition of satisfaction (Done Criteria) might include:
 - **Deliverables:** User guide, installer update
 - **Special activities:** such as: usability review
 - **Expected Behavior:** rainy cases
- Generic done criteria for typical activities
 - Coding : unit test Done, peer review, check-in
 - Design: review done

What makes a good story?

INVEST In Good Stories



Product Backlog

- A list of **all** desired work (stories), **Ordered by priority**
- Fluid list of functionality, technology, issues.
- Anyone can contribute.
- Contain placeholders for unclear issues for later refinement
- Organized by epics -> user stories -> sub-stories



Prioritized PBIs with Effort Estimates, Business Value, ROI and Earned Value

- > Effort Estimate = total story points (by team)
- > Business Value =
 - > Penalty (0-10) + Value (1-10)
 - > Spread 1000 value point between the stories
- > ROI = BV/Estimate
- > Earned Value of a feature = feature ROI / \sum ROIs in the release

Release Planning Process

- Top priority stories in low granularity (up to 20 days)
- Teams know their velocity
- Allocate stories to sprints **based on business value**
- Review your release plan – analyze **dependencies and constrains**
 - Challenge all technical dependencies !!!!!

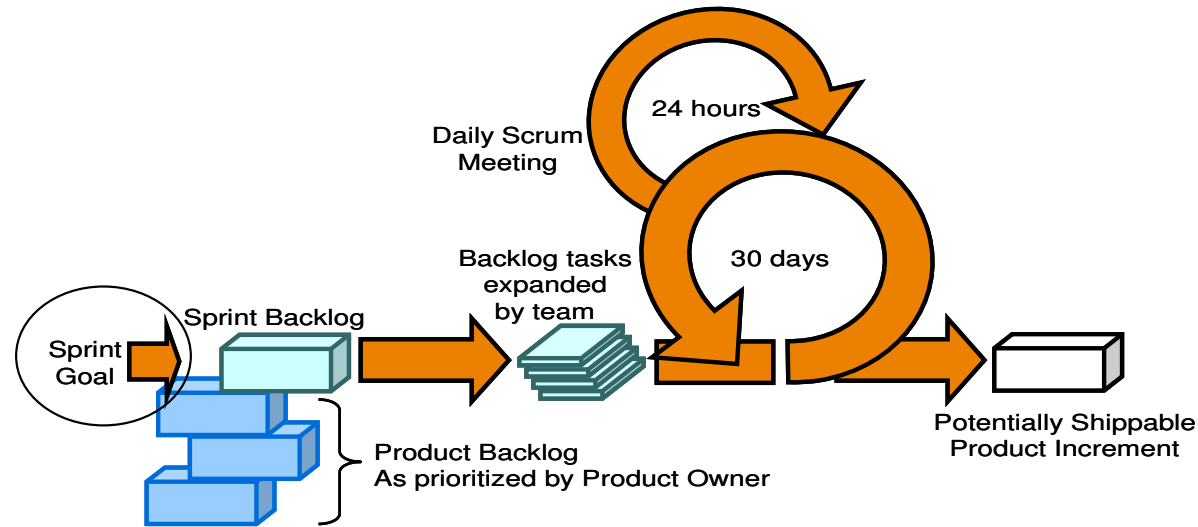
Release Planning Process (cont.)

- Challenge the plan
 - Am I asking too much for a sprint ?
 - Do I have too many topics ?
 - Special overload on specific person or skill ?
 - Do I have good utilization of people
 - Identify constrains and dependencies on “external” people/events
- Write good “elevator speech” for each sprint (Spring goal) and for the release
- Track and asses release plan every sprint !!!!

Sprint Planning

Sprint Goals

- Theme of the Sprint
- Each Sprint must include some business functionality, even the first sprint



Support
Features
necessary for
population
genetics studies

All high priority
legacy bugs
fixed

Rock-solid
stability on
three platforms
for production
release

Sprint Planning

Creating the Committed Backlog

- **Segment 1-** Product Owner describes the requirements, the user stories and acceptance criteria
- **Segment 2 -** Team decides what they COMMIT to deliver in the Sprint- up to 4 hours.
 - Team estimate sizing (planning poker).
 - **PO hear assumptions and Q&A**
 - **Negotiate with the team**
 - **Accept their decision !**

Sprint Execution

- Product owner available for questions
- Review intermediate work
- Involves in case of de-scoping or additional work

Sprint Review Meeting

- Team demonstrates what was built
 - Informality is encouraged. PowerPoint is discouraged
- PO declares whether the agreed Acceptance Criteria were met for each backlog item
- Uncompleted/unacceptable items returned to Product Backlog.
- New stories added to Product Backlog as necessary