Sprint planning Optimization in Agile Data Warehouse Design

Matteo Golfarelli
Stefano Rizzi
Elisa Turricchia

University of Bologna - Italy

14th International Conference on Data Warehousing and Knowledge Discovery (DaWaK'12)
September 03, 2012

Summary

- Motivating scenario
- Agile concepts
- Optimization model
- Model validation
- Summary and future work

Motivating scenario (1)

Problems

- The data warehouse design is long and complex
- Difficult to clearly assess the several factors affecting the data warehouse design (e.g., user needs, development constraints)

Side effects

- Wrong estimation
- Delays on delivery
- Dissatisfied customers

Motivating scenario (2)

Solution

- Making more flexible and faster the DW design applying agile principles
- Supporting the analysts during the planning phase

Our contribution

- An optimization model to support the DW planning problem with agile principles

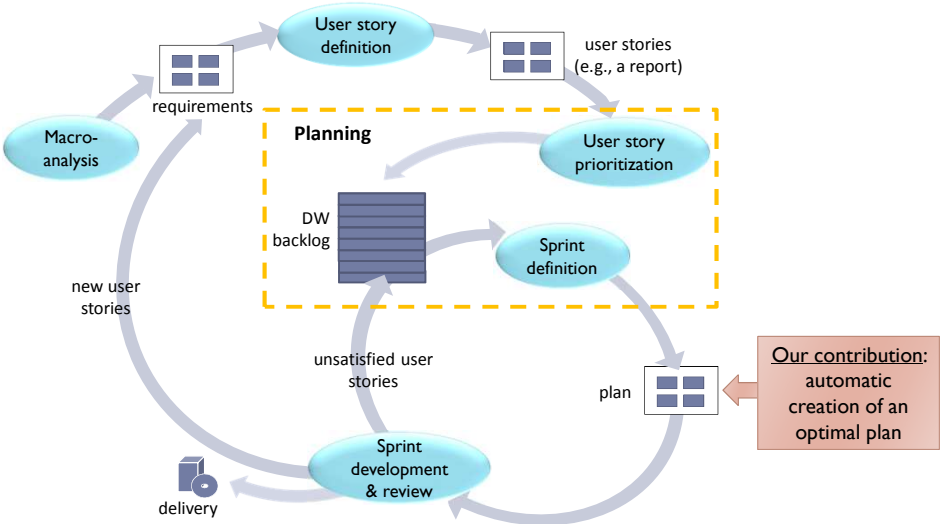
State of the art

- Agile data warehousing:
  - Scrum and eXtreme Programming in the DW context [1].
  - Four-Wheel-Drive (4WD): an agile design methodology for DW [2].
- Lack of optimization models for project scheduling that combine agile principles with DW features.
- A few tools for the agile project management (e.g., AgileFant [3], Mingle [4], ScrumWorks [5])

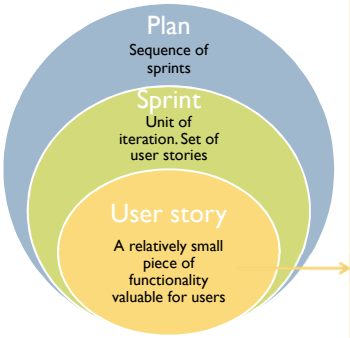
Agile data warehouse design practices [7,2]

- **Incremental process**: the DW system is broken up into smaller portions which are scheduled, developed, and integrated when completed.
- **Iteration**: the DW system is built in iterations, where each cycle expands the product until the project is completed.
- **User involvement**: continuous interaction with users is promoted to progressively refine the specifications.
- **Continuous and automated testing**: a DW is developed by refining and expanding an evolutionary prototype that progressively integrates the implementation of each increment.
- **Lean documentation**: small and simple formal schemata are preferred to extensive DW specifications.

Agile life-cycle for DW design



Optimization model: basic concepts (1)



User story features

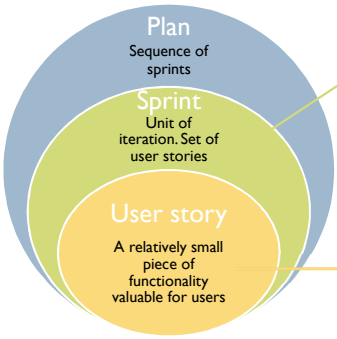
Utility: the business value of a user story (e.g., ranging from 10 to 100).

Story point: a unit of measurement for the development complexity of user stories (e.g., ranging from 1 to 10).

Risk: the risk that the project is not completed as desired.

- ❑**Critical story**: it has a strong impact on the other user stories, so that taking a wrong solution for it can dramatically affect the success of the project.
- ❑**Uncertain story**: is a story for which it is somehow hard to estimate the complexity due to unexpected problems that could arise.
- ❑**Class of risk**: no risk (1), low risk (1.3), medium (1.7), high risk (2)

Optimization model: basic concepts (2)



Sprint features

Duration: duration of a sprint in days.
Development speed: the estimated number of story points the team can deliver per day.

User story constraints

Affinity: the degree of correlation between user stories; similar stories have higher utility if they are included in the same sprint.
Dependence: a development constraint between two user stories, indicating that a user story (post-condition) cannot start before the other (pre-condition) is completed.

- ❑ **AND-type**: all the pre-condition stories must be completed.
- ❑ **OR-type**: at least one of the pre-condition stories must be completed.

Optimization model

Multi-knapsack problem [6]

- The knapsacks are the sprints and the items are the stories.
- The complexity (in story points) and the utility of an item represent its weight and value respectively.

Goals of an optimal plan

- **Customer satisfaction**: it can be obtained by delivering user stories with higher utility first.
- **Affinity management**: similar stories should be carried out in the same sprint to increase their value for users.
- **Risk management**:
  - ❑ Advancing critical user stories to avoid late side-effects.
  - ❑ Distributing uncertain stories in different sprints and postponing them to reduce the risk that the sprint delivery is delayed.

Sprint planning problem – Objective function (1)

$$z = Max \sum_{k=1}^{m} \sum_{i=1}^{k} \sum_{j=1}^{n} u_j \left(r_j^{cr} x_{ij} + a_j \frac{y_{ij}}{|Y_j|} \right)$$

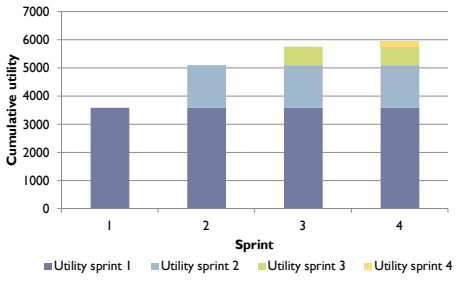
cumulative utility

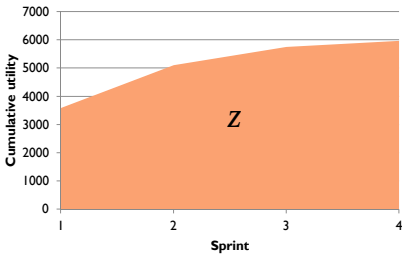
Affinity multiplier

m number of sprints;
n number of user stories;

$x_{ij} = 1$ iff story j is included in sprint i, 0 otherwise;
u_j utility of story j;
r_j^{cr} criticality risk of story j;
a_j affinity of story j;
U set of user stories;
$Y_j \subset U$ set of stories similar to story j;
y_{ij} accessory variable related to the number of stories in Y_j included in sprint i;

Sprint planning problem – Objective function (2)





- Advancing the stories with higher utility can increase objective function.
- The critical risk increases the utility of a story, encouraging an early placement of critical stories.
- The affinity increases the utility of a story proportionally to the fraction of similar stories included in the same sprint.

Sprint planning problem – Constraints (1)

$\sum_{j=1}^{n} p_j r_j^{un} x_{ij} \le p_i^{\max}$	$\forall i \in S$	The sum of the story points of the stories included in each sprint does not exceed the sprint capacity
$\sum_{i=1}^{m} x_{ij} = 1$	$\forall j \in U$	Each story is included in exactly one sprint
$\sum_{k=1}^{i} \sum_{z \in D_j} x_{kz} \ge x_{ij}$	$\forall i \in S, j \in U^{OR}$	OR dependence constraint
$\sum_{k=1}^{i} \sum_{z \in D_j} x_{kz} \ge x_{ij} \lvert D_j \rvert$	$\forall i \in S, j \in U^{AND}$	AND dependence constraint

Sprint planning problem – Constraints (2)

$y_{ij} \le \sum_{k \in Y_j} x_{ik}$	$\forall i \in S, j \in U$	Affinity management
$y_{ij} \le \lvert Y_j \rvert x_{ij}$	$\forall i \in S, j \in U$	

p_j complexity of story j;

r_j^{un} uncertain risk of story j;

$p_i^{\max}$ capacity of sprint i;

D_j dependences of story j;

U^{AND} subset of stories with AND-type dependences;

U^{OR} subset of stories with OR-type dependences;

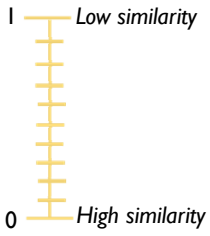
S set of sprints;

Model Validation: effectiveness tests

- **How to measure the distance between the optimal plan and the team plan?**

User story gap

$$gap(j) = \frac{1}{N-1}\left| i^{team} - i^{opt} \right|$$

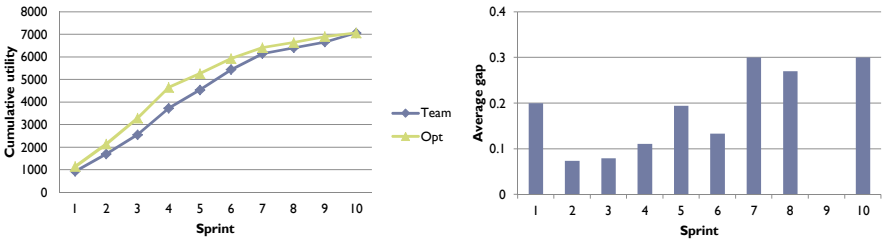


j	user story
i^{team}	is the sprint j belongs to in the team plan
i^{opt}	is the sprint j belongs to in the optimal plan
N	maximum number of sprints in the two plans

Model Validation: case study - 1

- Case study features
  - Pay-tv DW project
  - Duration: 8 months
  - # User stories: 44
  - # Sprints: 10 (with average duration of 17 days)
  - # Dependences: 52
  - Development speed: 2.43 story points per day

Model Validation: case study - 2

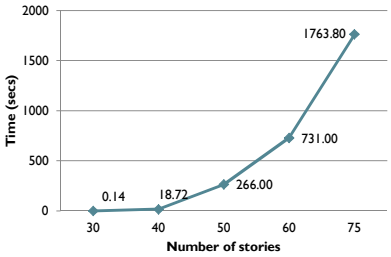


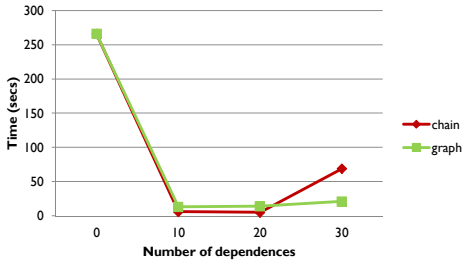
Comparison		
	Team plan	**Optimal plan**
Time to design a plan	Couple of days	Few seconds
Plan specification	Coarse estimations	Refined estimations
Risk distribution	Strong anticipation	More uniform distribution

Model Validation: efficiency tests – 1

- Benchmark
  - 58 synthetic projects
  - Utility values: [10,100]
  - Story point values: [1,10]
  - Sprint duration: 15 days
  - Development speed: 3 story points per day

Model Validation: efficiency tests – 2





- Exponential increase of the computation time.
- For complex problems (more than 100 stories), we can obtain an approximate solution (that is less than 1% worse than the optimal one) within 5 seconds.

- A small number of dependences (e.g., 10) tends to reduce the search space, reducing the computation time.
- A high number of dependences (e.g., 30) makes the problem more complex, increasing the computation time.

Summary and Future work

- We **formalize** the sprint planning problem for the agile DW design.
- We solve it with a **multi-knapsack model**.
- We carry out a **case study** and a set of tests on synthetic benchmarks to prove both effectiveness and efficiency of our approach.

..but we can extend our approach:

- Managing the plan evolution.
- Allowing different development velocity for different sprints.
- Modeling different team capability (e.g., design, implement, test).

References

[1] Hughes, R.: Agile Data Warehousing: Deliverng world-class business intelligence systems using Scrum and XP. Universe (2008).

[2] Golfarelli, M., Rizzi, S., Turricchia, E.: Modern software engineering methodologies meet data warehouse design: 4WD. In: Proc. DaWaK. Pp.66-79 (2011).

[3] Aalto University, SoberIT: Agilefant. http://www.agilefant.org/ (2011).

[4] ThoughtWorks Studios: Mingle: Agile project management. http://www.thoughtworks-studios.com/ (2011).

[5] Collabnet: ScrumWorks. http://www.danube.com/ (2011).

[6] Martello, S., Toth, P.: Knapsack Problems: Algorithm and Computer Implementation. John Wiley and Sons Ltd (1990).

[7] Dyba, T., Dingsoyr, T.: Empirical studies of agile software development: A systematic review. Information & Software Technology 50(9-10), 833-859 (2008).

Thank you for your attention

Questions?