COMTRADE

OSD-Model implementation on EOS-wnc

EOS Client for Windows

Gregor Molan

gregor@comtrade.com
gregor.molan@cern.ch

Topics

OSD-Model implementation on EOS-wnc

Introduction

It is the outlook of the theoretical background for EOS-wnc software development process management.

Background

Theoretical definitions and methodology

Graph theory

Definition 1. *An* **undirected graph** G *is a pair* $G = (V, E)$, *where* V *is a set of* **vertices** *(singular: vertex), and* E *is a set of* **edges**, *i.e. two-sets (set with two distinct elements) of vertices. The vertices* x *and* y *of an edge* $e = \overline{xy}$ *are called the* **endpoints** *of the edge. The edge* $e = \overline{xy}$ **joins** x *and* y *and it is* **incident** *on* x *and* y. *A vertex that does not belong to any edge is the* **singular vertex**. $\boxed{d}$

Lean software development

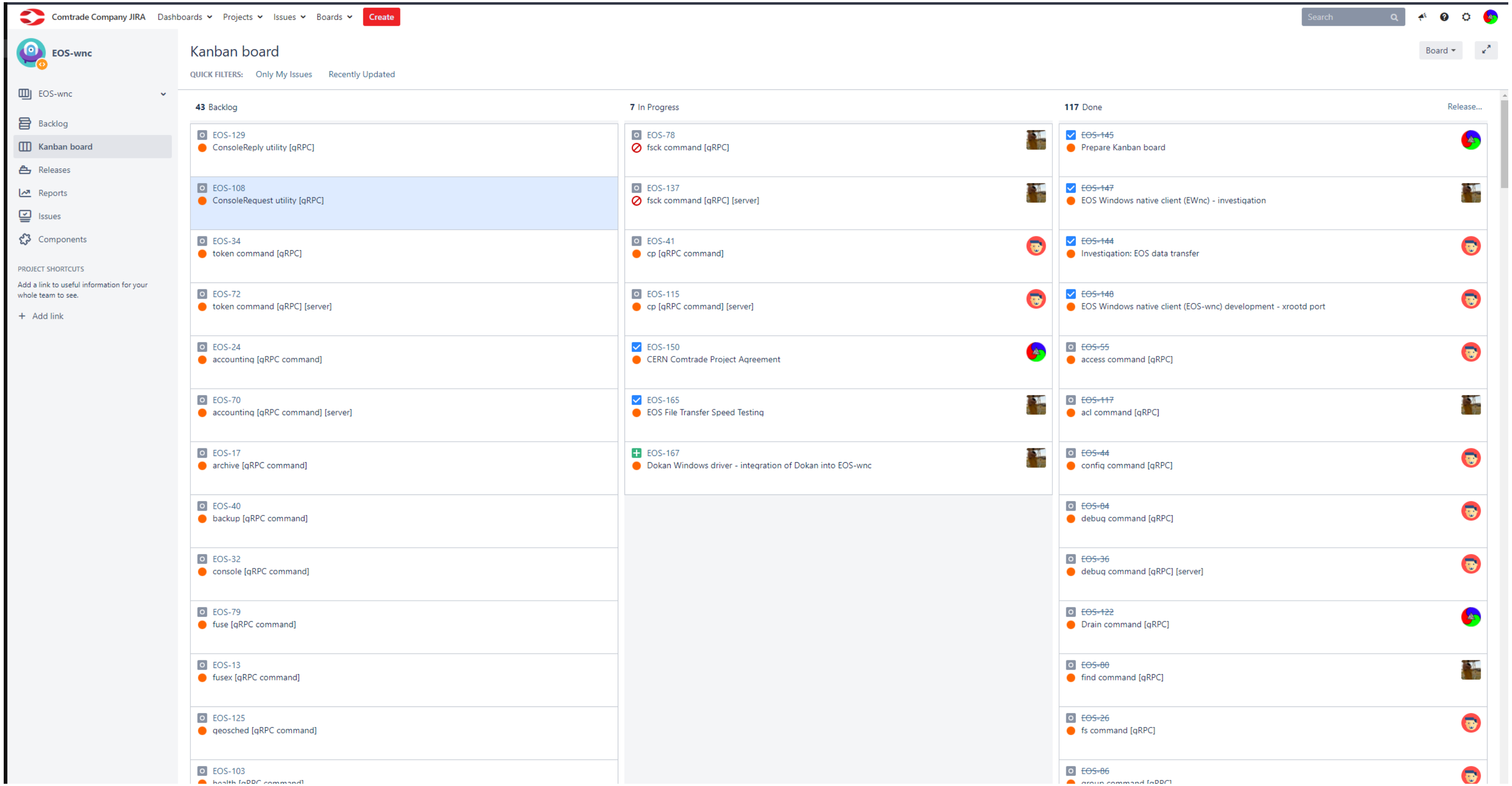
Follow simple principles:

1. Eliminate waste
2. Amplify learning
3. Decide as late as possible
4. Deliver as fast as possible
5. Empower the team
6. Build integrity in
7. Optimize the whole

Kanban method

- Upgrade of the lean software development
- Kanban boards
- Managing workflow
- Kanban metrics

JIRA project management tool



OSD-model

Functionality graph (f-graph)

Definition 2. *Let P be a software product. The* **functionality graph G^P** *(shortly* **f-graph***) of a software product P is a weighted undirected graph of functionalities and f-influences. Each vertex from the set of vertices in the f-graph G^P, which we denote also as $V(G^P)$, represents the functionality, and each edge from the set of edges in G^P, which we denote also as $E(G^P)$, represents the f-influence. The weights for vertices and edges in this graph are the following.*

π .. **the description of vertices and edges***:*

$$\pi : V(G^P) \cup E(G^P) \to \Pi_f \cup \Pi_i$$
$$\Pi_f = \{\text{“descr. of functionality } v_1\text{”}, \text{“descr. of functionality } v_2\text{”}, \ldots\}$$
$$\Pi_i = \{\text{“descr. of f-influence } e_1\text{”}, \text{“descr. of f-influence } e_2\text{”}, \ldots\}$$

δ .. **the development cost of vertices and edges***:*

$$\delta : V(G^P) \cup E(G^P) \to \mathbb{Z}_{\geq 0}, \quad e \in E(G^P),\ v \in V(G^P)$$
$\delta(v) = 0$ *if it is the devel. cost of existing functionality v*

$\delta(v) > 0$ *if it is the devel. cost of funct. v that should be developed*

$\delta(e) = 0$ *if it is the devel. cost of existing f-influence e*

$\delta(e) > 0$ *if it is the devel. cost of f-influence e*

$\tilde{\delta}$ .. **the development status of vertices and edges***:*

$$\tilde{\delta} : V(G^P) \cup E(G^P) \to \{0, 1\}, \quad e \in E(G^P),\ v \in V(G^P)$$
$$\tilde{\delta}(v) = \begin{cases} 0, & \text{if funct. } v \text{ should not be (or has been already) developed} \\ 1, & \text{if funct. } v \text{ should be developed} \end{cases}$$
$$\tilde{\delta}(e) = \begin{cases} 0, & \text{if f-influence } e \text{ should not be (or has been already) developed} \\ 1, & \text{if f-influence } e \text{ should be developed} \end{cases}$$

ϑ .. **the test cost of vertices and edges***:*

$$\vartheta : V(G^P) \cup E(G^P) \to \mathbb{Z}_{\geq 0}, \quad e \in E(G^P),\ v \in V(G^P)$$
$\vartheta(v)$ *is the test (unit test) cost of functionality v*

$\vartheta(e)$ *is the test cost of f-influence e*

$\tilde{\vartheta}$ .. **the test status of vertices and edges***:*

$$\tilde{\vartheta} : V(G^P) \cup E(G^P) \to \{0, 1\}, \quad e \in E(G^P),\ v \in V(G^P)$$
$$\tilde{\vartheta}(v) = \begin{cases} 0, & \text{if funct. } v \text{ should not be (or has been already) unit tested} \\ 1, & \text{if funct. } v \text{ should be unit tested} \end{cases}$$
$$\tilde{\vartheta}(e) = \begin{cases} 0, & \text{if f-influence } e \text{ should not be (or has been already) tested} \\ 1, & \text{if f-influence } e \text{ should be tested} \end{cases}$$

σ .. **the significance weight of vertices and edges***:*

$$\sigma : V(G^P) \cup E(G^P) \to [0, 1], \text{ where } \sum_{v \in V(G^P)} \sigma(v) = 1 \text{ and } \sum_{e \in E(G^P)} \sigma(e) = 1 \quad (1)$$

λ .. **the implementation cost of vertices and edges***:*

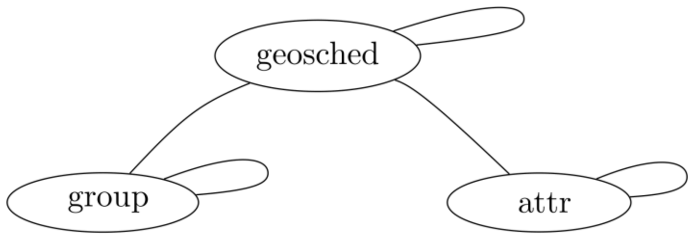
$$\lambda : V(G^P) \cup E(G^P) \to \mathbb{Z}_{\geq 0}, \quad e \in E(G^P),\ v \in V(G^P)$$
$$\lambda(v) = \delta(v)\,\tilde{\delta}(v) + \vartheta(v)\,\tilde{\vartheta}(v) \text{ and } \lambda(e) = \delta(e)\,\tilde{\delta}(e) + \vartheta(e)\,\tilde{\vartheta}(e) \quad (2)$$

ϵ .. **the implementation value of vertices and edges***:*

$$\epsilon : V(G^P) \cup E(G^P) \to [0, 1], \quad e \in E(G^P),\ v \in V(G^P)$$
$$\epsilon(v) = \tilde{\delta}(v)\,\tilde{\vartheta}(v)\,\sigma(v) \text{ and } \epsilon(e) = \tilde{\delta}(e)\,\tilde{\vartheta}(e)\,\sigma(e) \quad (3)$$

Significance weight defined in (1) is defined as a probability density function restricted to $V(G^P)$ or $E(G^P)$ and therefore has values in $[0, 1]$. Significance weights $\sigma(v)$ and $\sigma(e)$ for functionality v and f-influence e are estimated relative to other functionalities and f-influences regarding to their relative importance from a customer's point of view. We assume that values for significance weights functions are defined and its definitions are not scope of this article.

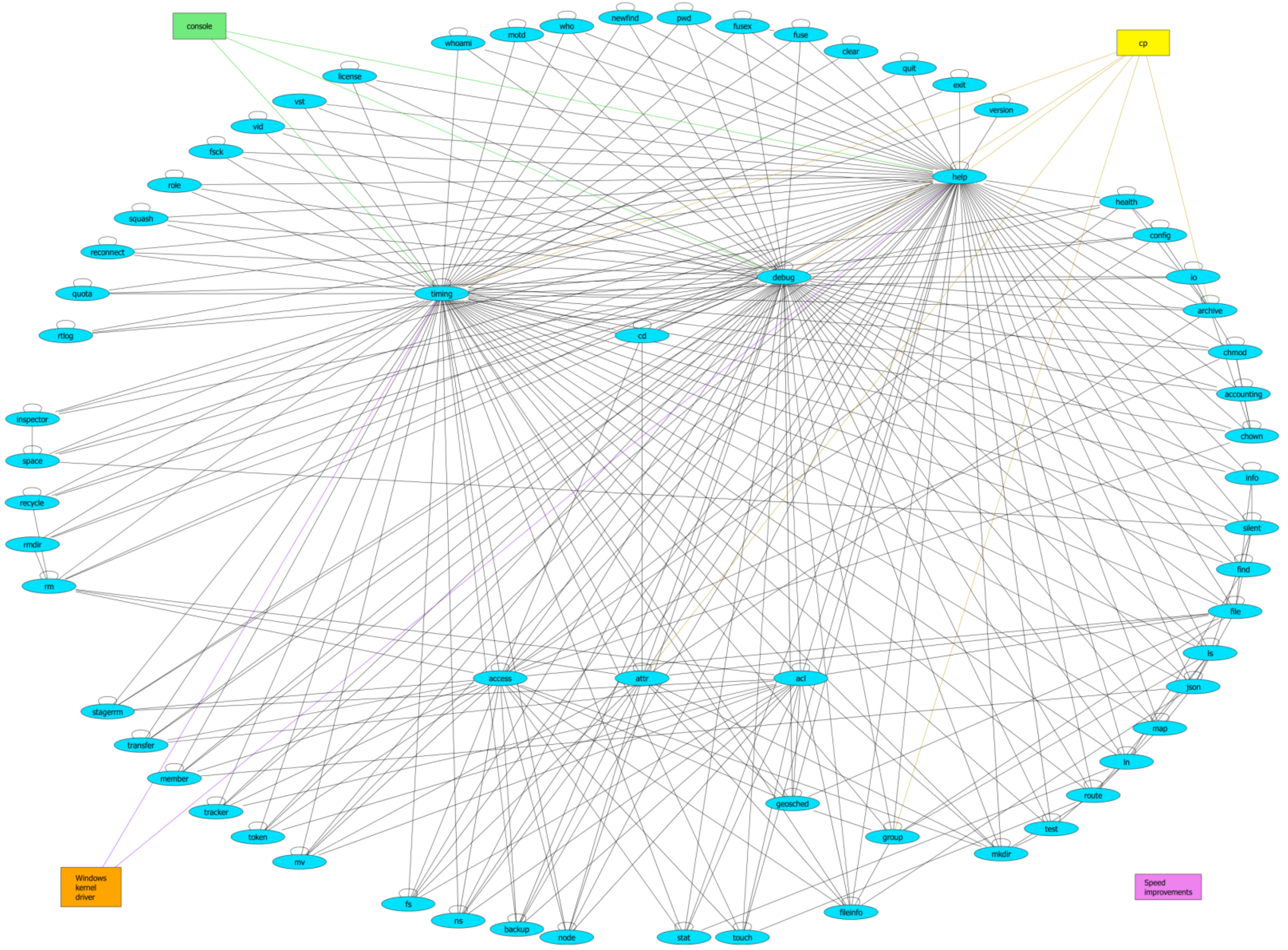
Example of the f-graph



Functionality	The functionality presented as a graph weight value $\pi(v)$
group	EOS function group
goesched	EOS function goesched
attr	EOS function attr

f-influence	The f-influence presented as a graph weight value $\pi(e)$
$group - group$	stand-alone testing of group function
$group - goesched$	testing of connections between functions group and geosched
$geosched - goesched$	stand-alone testing of group geosched
$goesched - attr$	testing of connections between functions geosched and attr
$attr - attr$	stand-alone testing of attr function

f-graph for the EOS-wnc

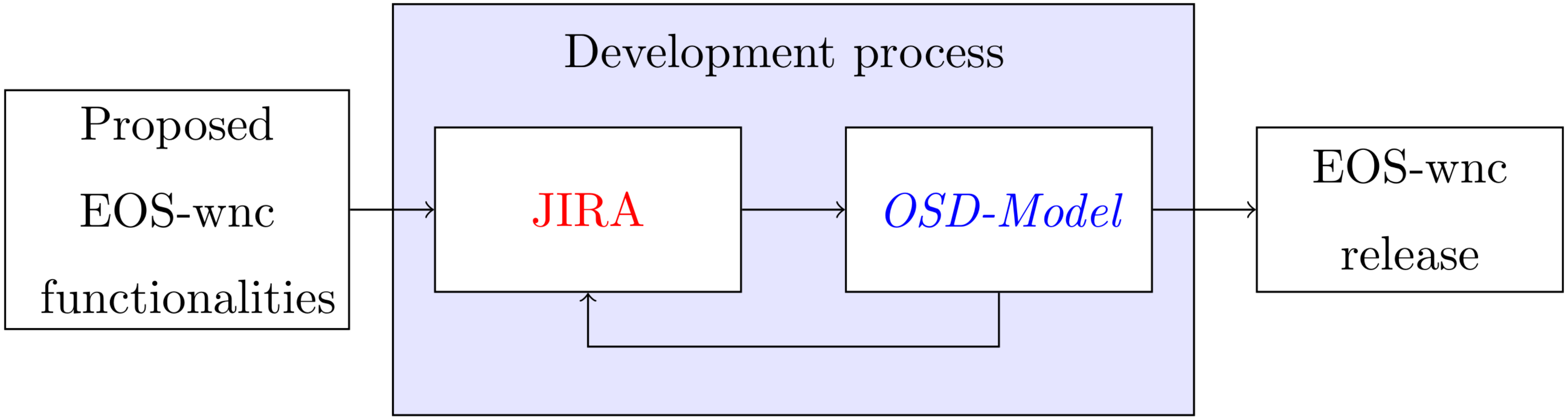


COMTRADE

2021-03-02

Gregor Molan

JIRA integration



Implementation phase

Definition 4. *Let G^P be the f-graph. The* **test suite** *of the f-graph G^P is a subgraph $H \subseteq G^P$, which does not have isolated vertices, if number of vertices $|V(H)| > 1$. Test suite also defines values for test status weight $\tilde{\vartheta}()$ and values for development status weight*[1] *$\tilde{\delta}()$ for all vertices and edges from this test suite.*

Definition 5. *Let G^P be the f-graph. An* **implementation phase $\mathcal{H}$** *of the f-graph G^P is the finite sequence of n test suites that covers all vertices and all edges from the f-graph:*

$$\mathcal{H} = \{{}^iH\}_{i=1}^n, \text{ where } \bigcup_{i=1}^n V({}^iH) = V(\mathcal{H}) = V(G^P) \quad \text{and} \quad \bigcup_{i=1}^n E({}^iH) = E(\mathcal{H}) = E(G^P). \tag{11}$$

Let us write $V({}^iH)$ for a set of all vertices in the test suite ${}^iH$ and $V(\mathcal{H})$ for a set $\cup_{i=1}^n V({}^iH)$. Analogously for sets of edges $E({}^iH)$ and $E(\mathcal{H})$. For a test suite ${}^iH$ we call index $i \in \mathbb{Z}^+$ a **test suite identifier of** *${}^iH$. According to Definition 4 the subgraph presents the test suite in a testing process.*

OSD algorithms

Optimum software development (OSD)

Definition 7. *Let* G^P *be the f-graph,* δ_T *the development cost for a trivial implementation phase, and let* $W \in \mathbb{Z}_{\geq 0}$ *with* $W \geqslant \delta_T$*. The* **optimal implementation phase** *for* G^P *and for given* W *is the implementation phase* $\boldsymbol{\mathcal{H}_{opt} = OIP(G^P, W)}$*, such that* $\lambda(\mathcal{H}_{opt}) \leqslant W$ *and*

$$\forall \mathcal{H} \;:\; \big(\epsilon(\mathcal{H}) > \epsilon(\mathcal{H}_{opt})\big) \;\Rightarrow\; \big(\lambda(\mathcal{H}) > \lambda(\mathcal{H}_{opt})\big). \qquad \boxed{d} \quad (12)$$

OSD algorithms

$$
\begin{aligned}
\mathit{maximize}\ \ & \epsilon(\mathcal{H}_{\tilde{\vartheta}_S \tilde{\delta}_T}),\\
\mathit{subject\ to}\ \ & \lambda(\mathcal{H}_{\tilde{\vartheta}_S \tilde{\delta}_T}) \leq W,\\
& S \subseteq E(G^P),\ \tilde{\vartheta}_S(S) = 0,\\
& T \subseteq V(G^P),\ \tilde{\delta}_T(T) = 0.
\end{aligned}
$$

Conclusion and future work

Theoretical background is successfully used in EOS-wnc software development process.

Future work: Software integration of OSD algorithms with JIRA.