GUI für EEROS Sicherheits- und Kontrollsystem

Artan Papaj

Inhalt

- Ausgangslage
- Konzept Safety System
- Informationen extrahieren
- Weitere Arbeit

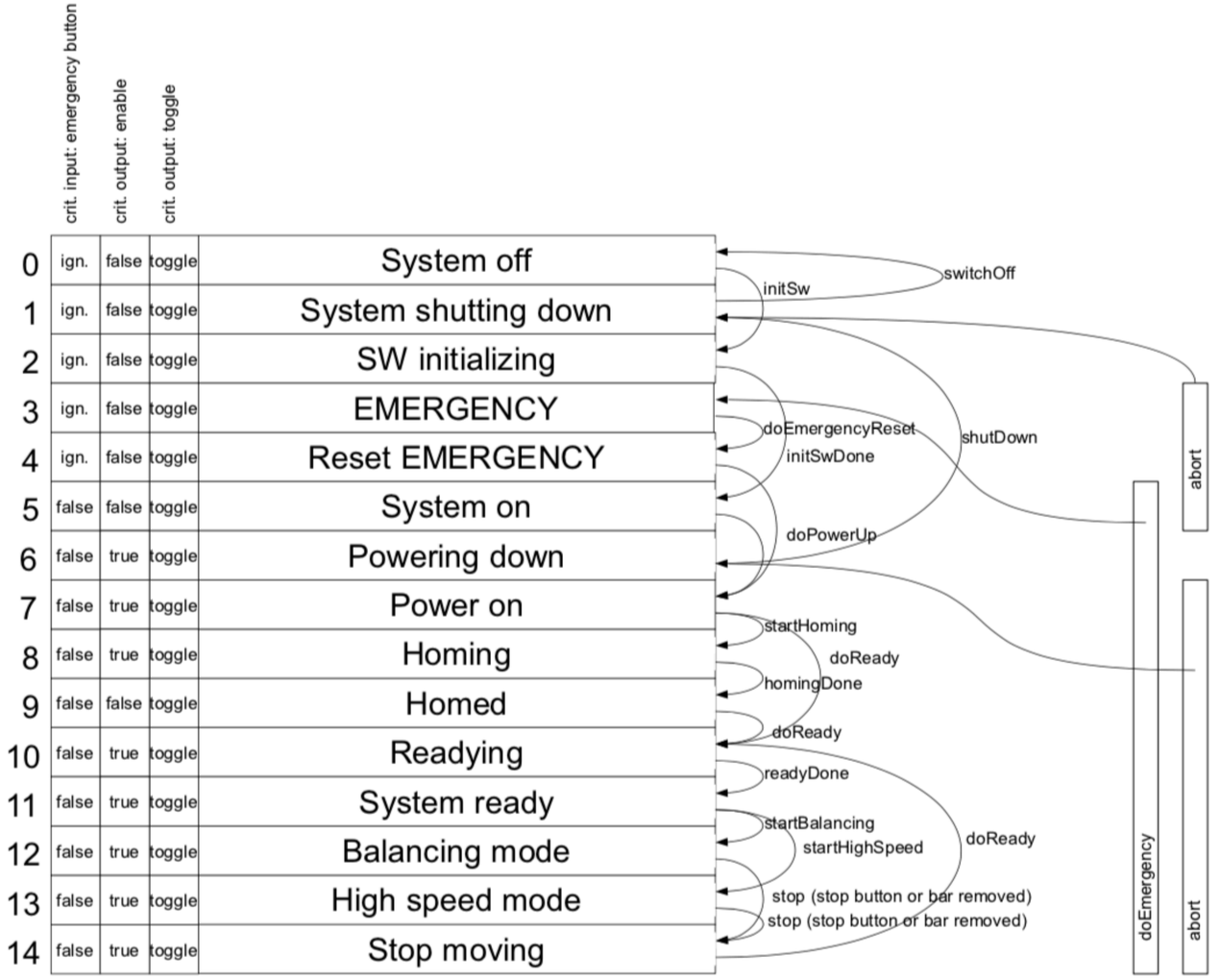
Ausgangslage

- EEROS
- Kontroll- und Sicherheitssystem in Code
- Intuitive Visualisierung und GUI

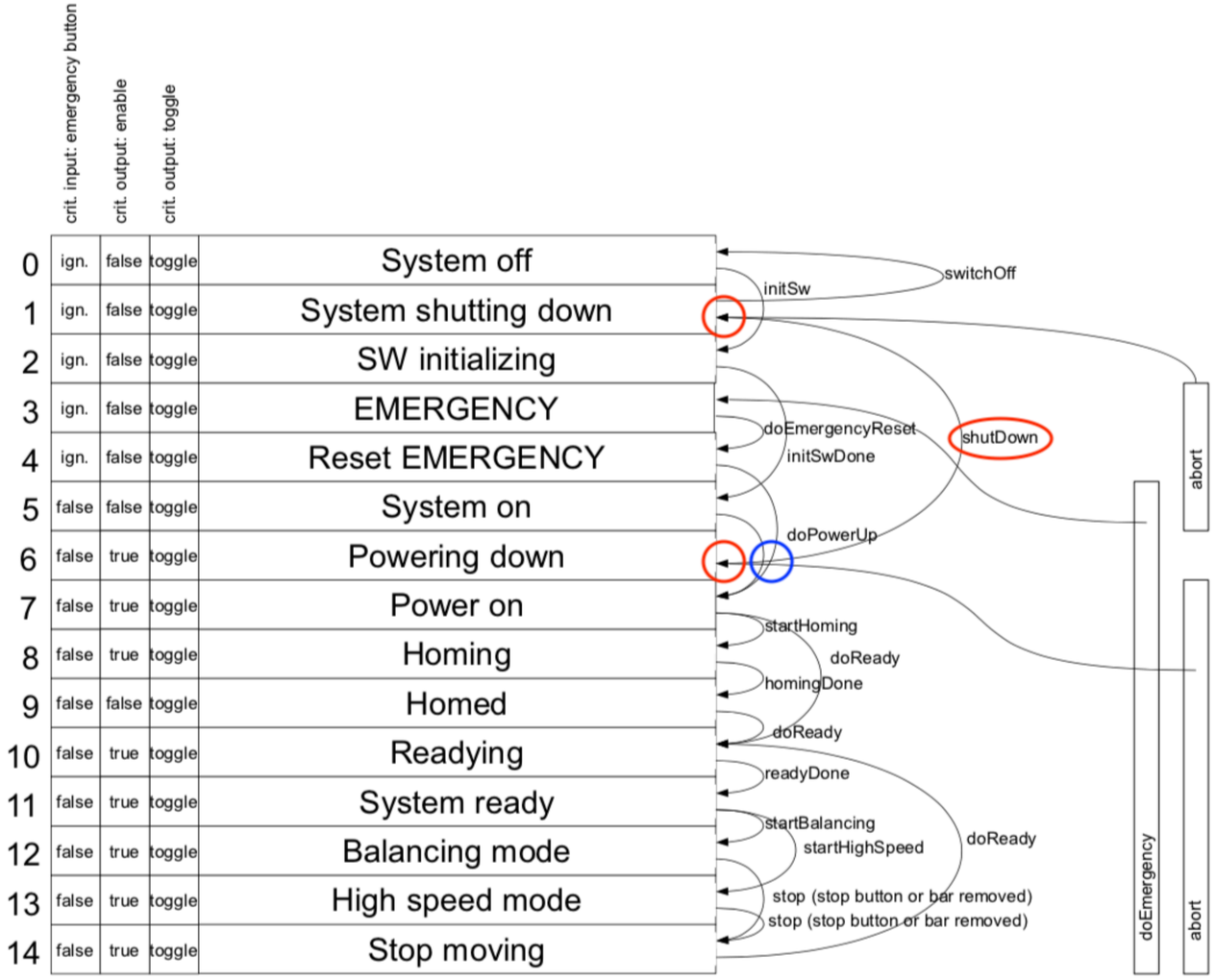
Zu visualisierende Informationen Safety System

- Safety Levels und deren Hierarchie
- Safety Events
- Übergänge von einem Safety Level zu einem anderen durch Safety Events
- Critical Inputs und Critical Outputs und deren Werte

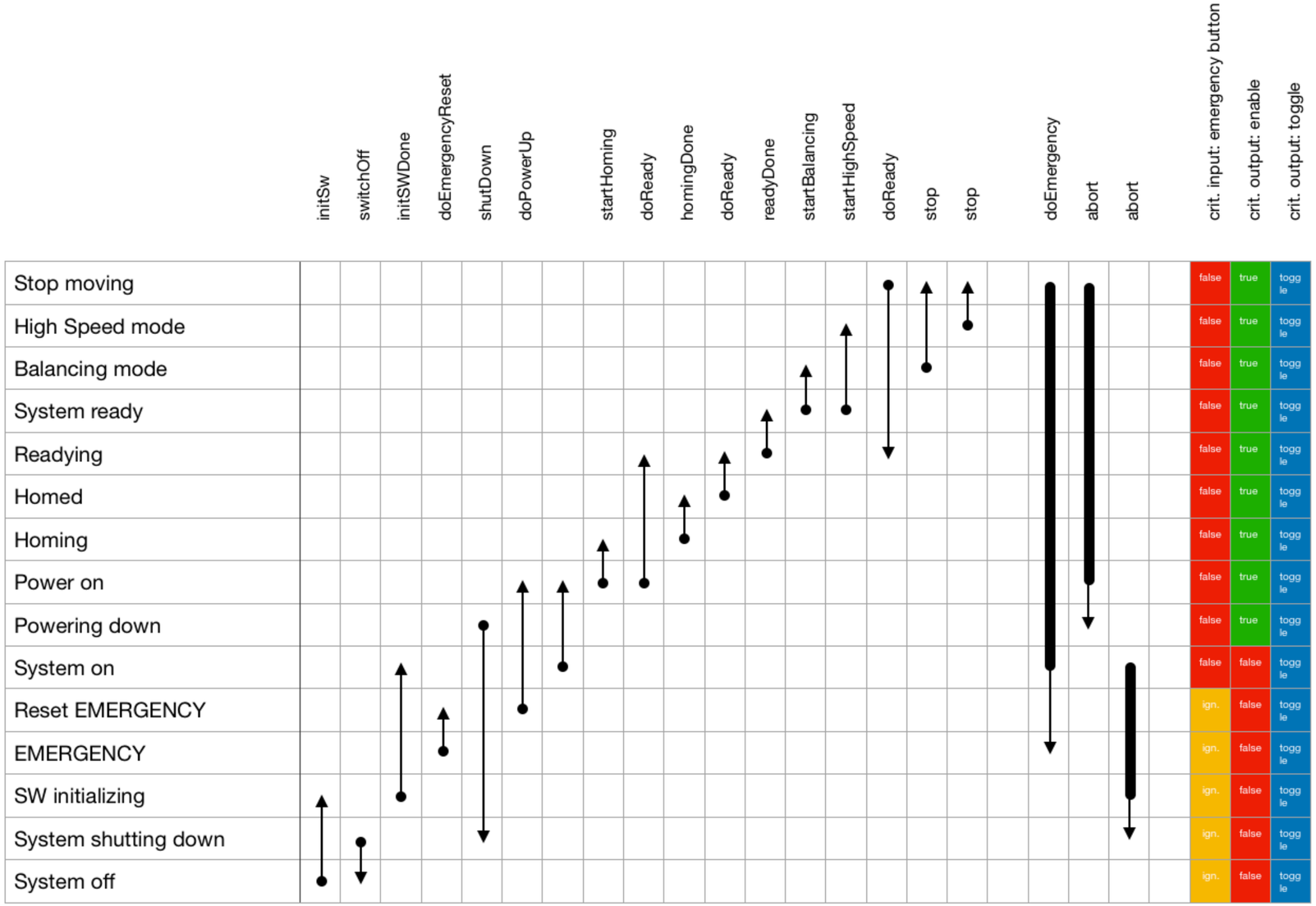
Ursprüngliche Visualisierung



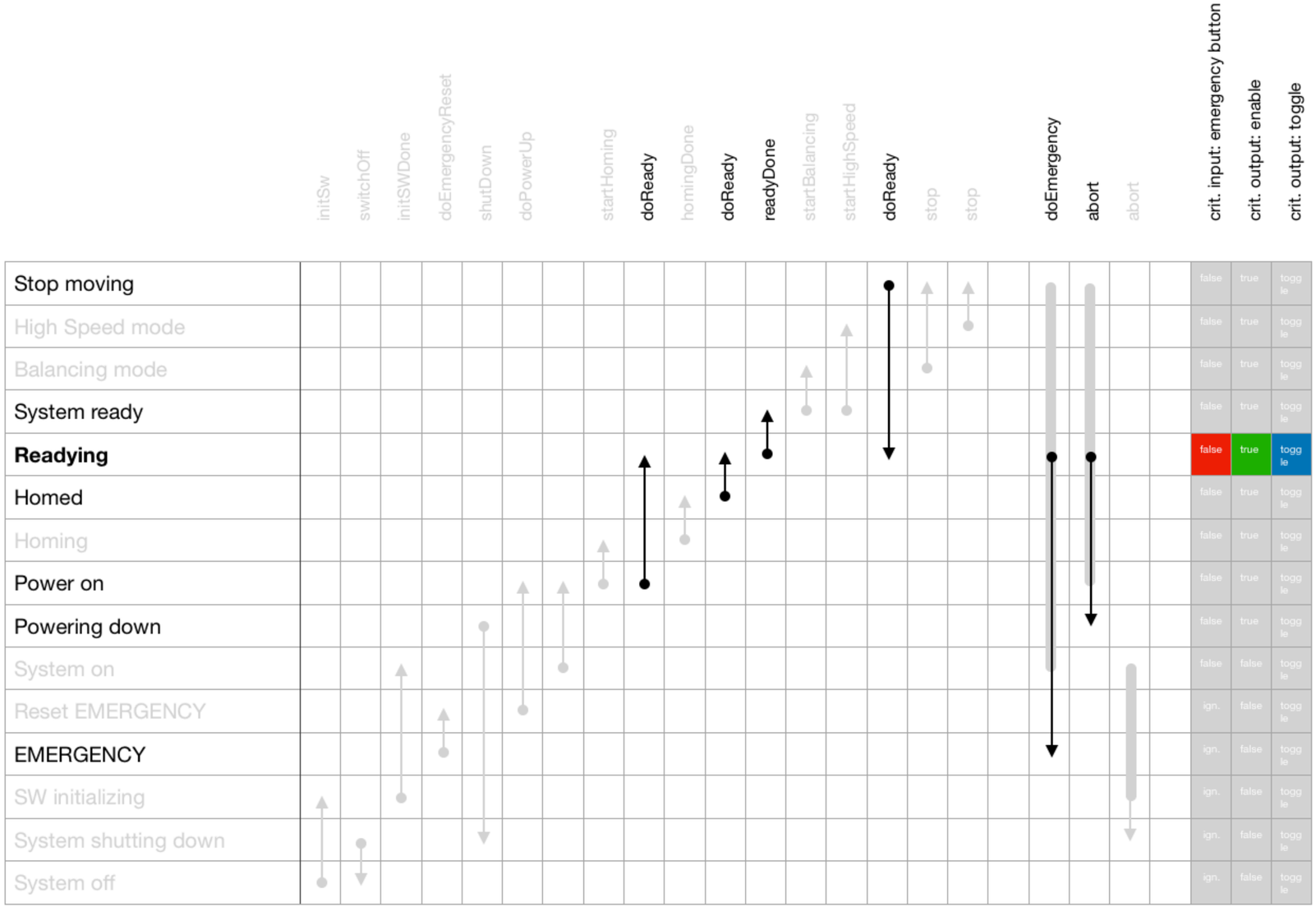
Ursprüngliche Visualisierung



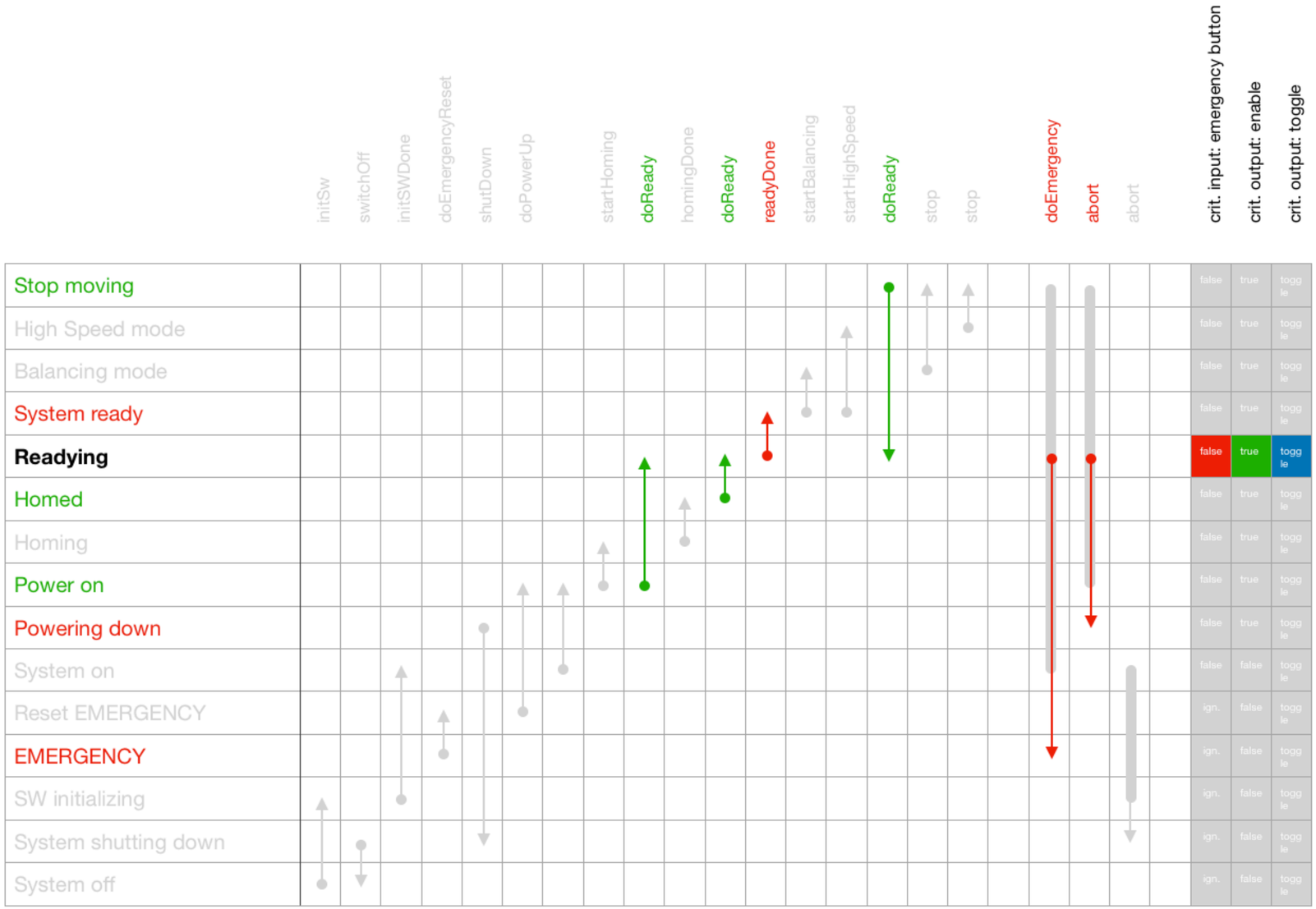
Konzept Safety System



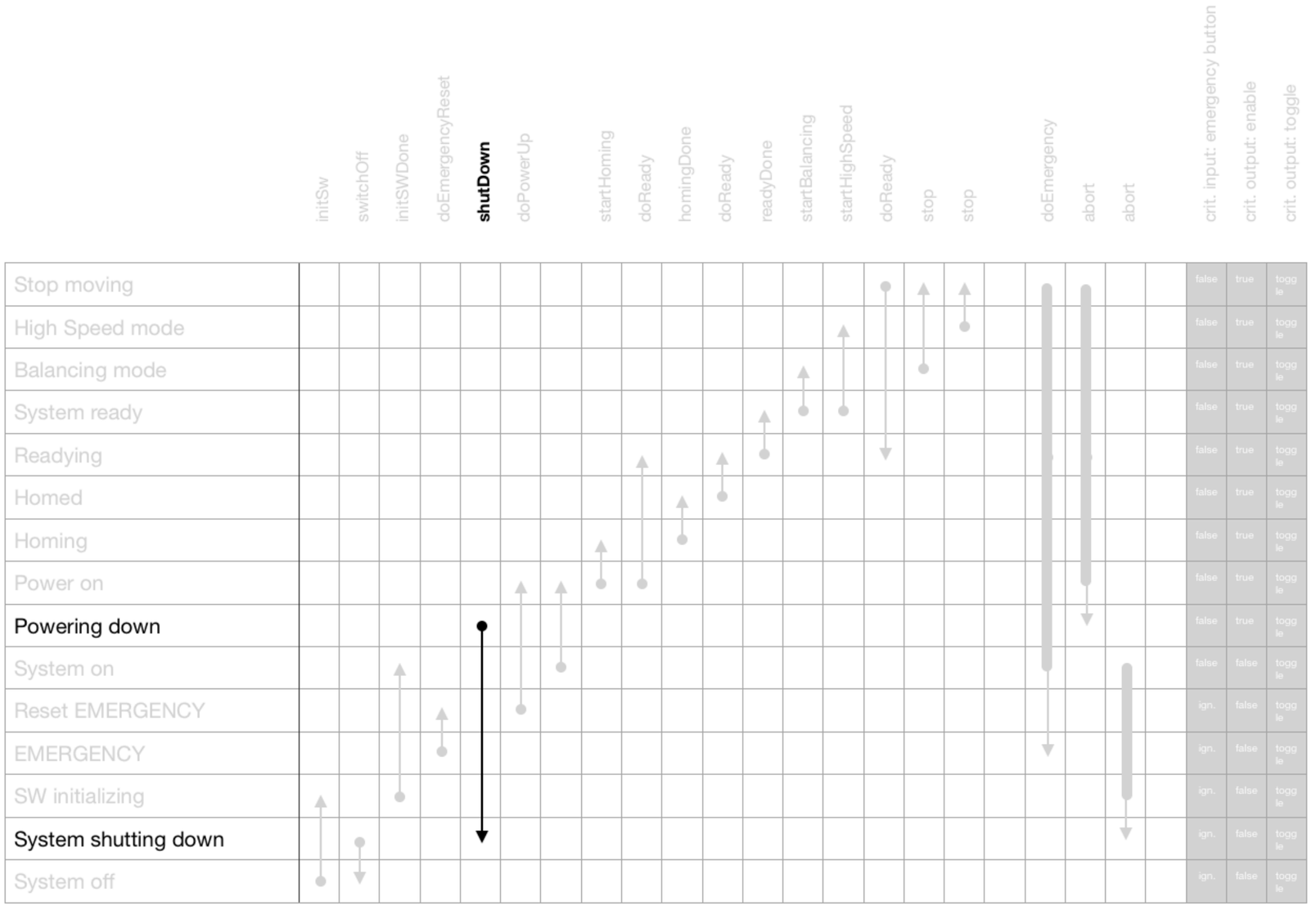
Auswahl eines Safety Levels



Auswahl eines Safety Levels mit Farbe



Auswahl eines Safety Events



Vorteile Konzept Safety System

- Richtungen der Safety Events klar
- Keine Kreuzungen der Pfeile
- Einheitlich
- Leicht Lesbar
- Fokus auf Safety Levels oder Safety Events möglich

Modifikation in der GUI

- Pfeile zeichnen
- Zeilen und Spalten hinzufügen oder löschen
- Toggeln

Informationen extrahieren

- Code Instrumentieren
- Reflexion in C++
- KDevelop API

Benötigte Informationen

- Safety Levels und deren Hierarchie
- Safety Events
- Ausgangs-Safety Level und Eingangs-Safety Level eines Safety Events
- Critical Inputs und Critical Outputs und deren Werte

Code Instrumentieren

```
// ############ Add levels ############
addLevel(slOff);
addLevel(slShuttingDown);
addLevel(slIinitializing);
addLevel(slInitialized);
addLevel(slRunning);

// ############ Add events to the levels ############
slOff.addEvent(seStartInitializing, slIinitializing, kPublicEvent);
slShuttingDown.addEvent(seSwitchingOff, slOff, kPrivateEvent);
slIinitializing.addEvent(seInitializingDone, slInitialized, kPublicEvent);
slInitialized.addEvent(seStartRunning, slRunning, kPublicEvent);
slRunning.addEvent(seStopRunning, slInitialized, kPrivateEvent);
addEventToLevelAndAbove(slIinitializing, seShutDown, slShuttingDown, kPublicEvent);

// ############ Define input states and events for all levels ############
slOff.setInputAction({ ignore(in1) });
slShuttingDown.setInputAction({ ignore(in1) });
slIinitializing.setInputAction({ ignore(in1) });
slInitialized.setInputAction({ check(in1, false, seStartRunning) });
slRunning.setInputAction({ check(in1, true, seStopRunning) });
```

Code Instrumentieren

```
void SafetyLevel::addEvent(SafetyEvent event, SafetyLevel& nextLevel, EventType type) {
    transitions.insert(std::make_pair(event.id, std::make_pair(&nextLevel, type)));
    std::cout << "event: '" << event << "' from '" << this << "' to '" << nextLevel << "'" << std::endl;
}
```

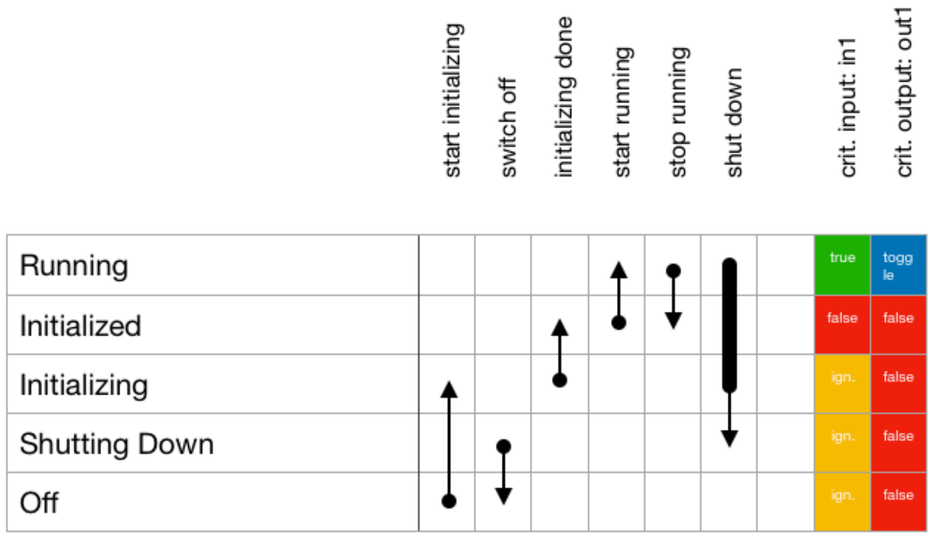
```
// overloading << operator for inputAction
std::ostream& operator<<(std::ostream& os, eeros::safety::InputAction* inputAction) {
    eeros::safety::IgnoreInputAction<bool>* iia = dynamic_cast<IgnoreInputAction<bool>*> (inputAction);
    eeros::safety::CheckInputAction<bool>* cia = dynamic_cast<CheckInputAction<bool>*> (inputAction);
    eeros::safety::CheckRangeInputAction<bool>* cria = dynamic_cast<CheckRangeInputAction<bool>*> (inputAction);
    eeros::safety::InputAction* ia = dynamic_cast<InputAction*> (inputAction);

    if(iia) {
        os << "'ignore' input '" << iia->getInput()->getId() << "'";
    } else if(cia) {
        os << "'check' input '" << cia->getInput()->getId() << "' with value '" << cia->getValue() << "'";
    } else if(cria) {
        os << "'check range' of input '" << cria->getInput()->getId() << "'";
    } else if (ia) {
        os << "of type InputAction";
    }
    return os;
}
```

Code Instrumentieren

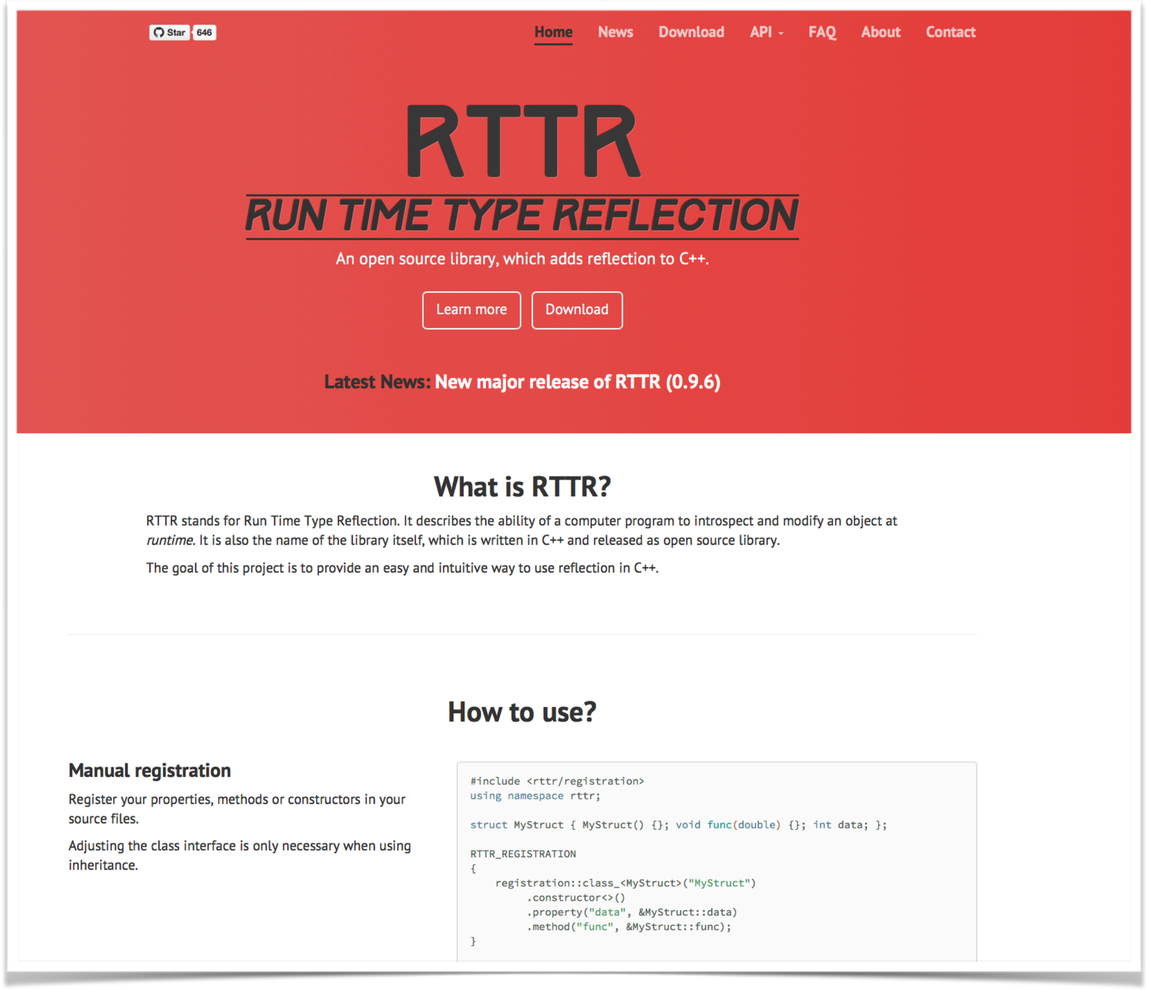
```
ntb@ImageINF2018:~/projects/eeros/build/examples/safety$ ./safetySystemTest1 -c SafetySystemTest1Config.json
2018-08-29 17:58:16    I:  Safety System Example started...
level 1: 'off'
level 2: 'shutting down'
level 3: 'initializing'
level 4: 'initialized'
level 5: 'running'
event: 'start initializing' from 'off' to 'initializing'
event: 'switching off' from 'shutting down' to 'off'
event: 'initialization done' from 'initializing' to 'initialized'
event: 'start running' from 'initialized' to 'running'
event: 'stop running' from 'running' to 'initialized'
event: 'start shutting down' from 'initializing' to 'shutting down'
event: 'start shutting down' from 'initialized' to 'shutting down'
event: 'start shutting down' from 'running' to 'shutting down'
input action: 'ignore' input 'in1' of level 'off'
input action: 'ignore' input 'in1' of level 'shutting down'
input action: 'ignore' input 'in1' of level 'initializing'
input action: 'check' input 'in1' with value '0' of level 'initialized'
input action: 'check' input 'in1' with value '1' of level 'running'
output action: 'set' input 'out1' with value '0' of level 'off'
output action: 'set' input 'out1' with value '0' of level 'shutting down'
output action: 'set' input 'out1' with value '0' of level 'initializing'
output action: 'set' input 'out1' with value '0' of level 'initialized'
output action: 'toggle' ouput 'out1' of level 'running'
2018-08-29 17:58:16  S W:  safety system verified: 5 safety levels are present
2018-08-29 17:58:16  S I:  triggering event 'start initializing' in level 'off': transition to safety level: 'initializing'
2018-08-29 17:58:16  A E:  could not set realtime priority
2018-08-29 17:58:16  A E:  could not lock memory in RAM
2018-08-29 17:58:17  E E:  could not set realtime priority
2018-08-29 17:58:17  E E:  could not lock memory in RAM
2018-08-29 17:58:23  S I:  triggering event 'initialization done' in level 'initializing': transition to safety level: 'initialized'
^C2018-08-29 17:58:24  S I:  triggering event 'start shutting down' in level 'initializing': transition to safety level: 'shutting down'
2018-08-29 17:58:24  S I:  triggering event 'switching off' in level 'shutting down': transition to safety level: 'off'
2018-08-29 17:58:25    I:  Test finished...
```

Code Instrumentieren



Reflexion in C++

- RTTI (Run-Time Type Information)
- Eigene Implementation
- Open Source Bibliothek



KDevelop API

KDE API Reference

KDE API Reference kdevelop API Reference KDE Home Contact Us

kdevelop/kdevplatform/language/highlighting

- Main Page
- Namespace List
- Namespace Members
- Alphabetical List
- Class List
- Class Hierarchy
- Class Members
- File List
- File Members
- Related Pages

kdevelop API Reference

- kdevplatform
  - debugger
  - documentation
  - interfaces
  - language
    - assistant
    - backgroundparser
    - checks
    - classmodel
    - codecompletion
    - codegen
    - duchain
    - editor
    - highlighting
    - interfaces
    - util
  - outputview
  - project
  - serialization
  - shell
  - sublime
  - tests
  - util
  - vcs

Search

search term

ALL ALL

Go

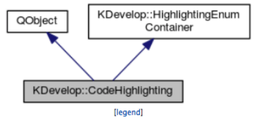
KDevelop CodeHighlighting

Public Member Functions | Protected Member Functions | List of all members

KDevelop::CodeHighlighting Class Reference

`#include <codehighlighting.h>`

Inheritance diagram for KDevelop::CodeHighlighting:



Public Member Functions

	CodeHighlighting (QObject *parent)
	~CodeHighlighting () override
KTextEditor::Attribute::Ptr	**attributeForDepth** (int depth) const
KTextEditor::Attribute::Ptr	**attributeForType** (Types type, Contexts context, const QColor &color) const
bool	**hasHighlighting** (IndexedString url) const override
void	**highlightDUChain** (ReferencedTopDUContext context) override

▸ Public Member Functions inherited from **QObject**

Protected Member Functions

virtual **CodeHighlightingInstance** * **createInstance** () const

▸ Protected Member Functions inherited from **QObject**

Additional Inherited Members

- ▸ Public Types inherited from **KDevelop::HighlightingEnumContainer**
- ▸ Static Public Member Functions inherited from **QObject**
- ▸ Static Protected Member Functions inherited from **QObject**
- ▸ Properties inherited from **QObject**

Detailed Description

General class representing the code highlighting for one language.

Definition at line 144 of file codehighlighting.h.

Constructor & Destructor Documentation

KDevelop::CodeHighlighting::CodeHighlighting (QObject * **parent**) **explicit**

Todo:
Don't highlighting everything, only what is visible on-demand

Definition at line 60 of file codehighlighting.cpp.

KDevelop::CodeHighlighting::~CodeHighlighting ( ) **override**

Definition at line 71 of file codehighlighting.cpp.

Weitere Arbeit

- C++ Reflexion Bibliothek
- KDevelop API
- Umsetzung der Modifikation an der Repräsentation
- Implementation