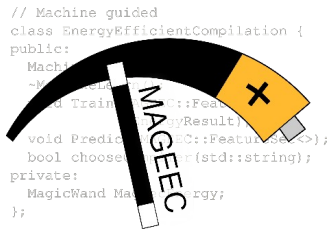


# Machine Guided Energy Efficient Compilation

Jeremy Bennett & Simon Cook, Embecosm

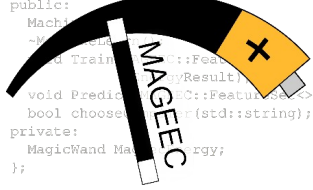


# What is MAGEEC?



Today we optimize for  
speed or space

```
// Machine guided
class EnergyEfficientCompilation {
public:
    MagicWand Wand;
    ~MagicWand() { Wand.Train(); }
    void Predict(C::FeatureSet &fs, C::Result &r);
    void Choose(C::FeatureSet &fs, C::Result &r);
private:
    MagicWand Wand;
};
```



# What is MAGEEC?

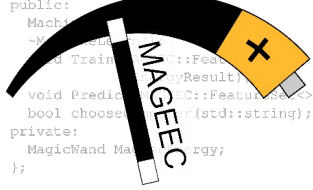


Today we optimize for  
speed or space



What if we could optimize for  
energy usage?

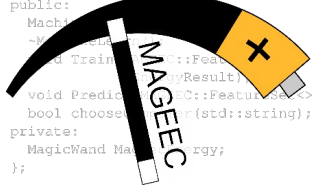
```
// Machine guided
class EnergyEfficientCompilation {
public:
    Machi
    ~Mach
    void Train(C::Feat
    void Predict(C::Feat
    bool choose(std::string);
private:
    MagicWand Ma
};
```



Research into modeling  
energy usage

# How We Got Here

```
// Machine guided
class EnergyEfficientCompilation {
public:
    Machi
    ~Mach
    void Train(C::Feat
    void Predict(C::Feat
    bool choose(std::string);
private:
    MagicWand Ma
};
```



Research into modeling  
energy usage



Energy  
measurement

# How We Got Here

```
// Machine guided
class EnergyEfficientCompilation {
public:
    Machi
    ~Mach
    void Train(C::Fea
    void Predict(C::Fea
    bool choose(C::Fea
private:
    MagicWand Ma
};
```



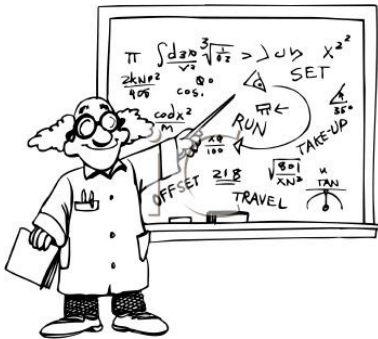
# How We Got Here



Research into modeling  
energy usage

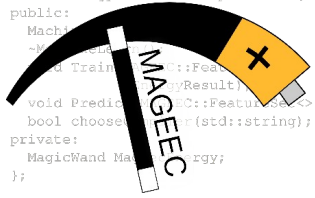


Energy  
measurement



Research into feedback  
directed optimization

```
// Machine guided
class EnergyEfficientCompilation {
public:
    Machi
    ~Mach
    void Train(C::Feat
    void Predict(C::Feat
    bool choose(C::std::string);
private:
    MagicWand Ma
};
```



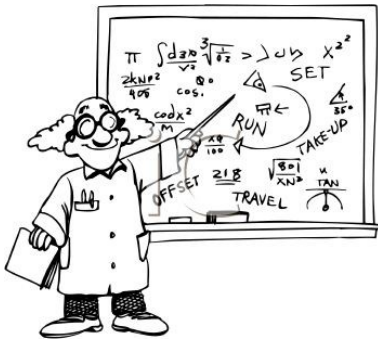
# How We Got Here



Research into modeling  
energy usage



Energy  
measurement

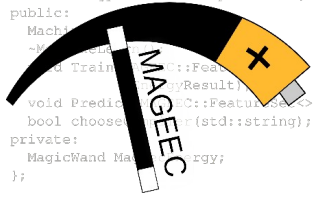


Research into feedback  
directed optimization





```
// Machine guided
class EnergyEfficientCompilation {
public:
    MagicWand Wand;
    void Train(C::FeatureSet<T> Result);
    void Predict(C::FeatureSet<T>);
    bool choose(std::string);
private:
    MagicWand Wand;
};
```



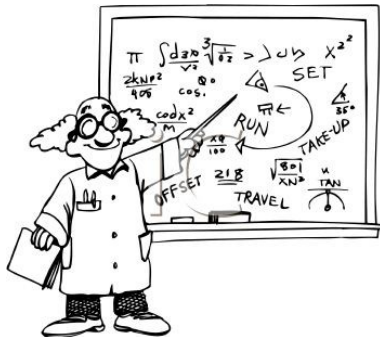
# How We Got Here



Research into modeling  
energy usage



Energy  
measurement



Research into feedback  
directed optimization





```
// Machine guided
class EnergyEfficientCompilation {
public:
    MagicWand Wand;
    ~MagicWand() {
        void Predict(MagicWand::FeatureSet<>);
        bool choose(MagicWand::FeatureSet<>);
    private:
        MagicWand MagicWand;
    };
};
```

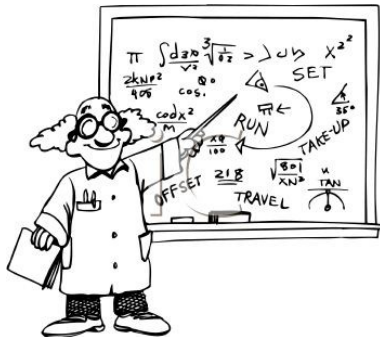
# How We Got Here



Research into modeling  
energy usage



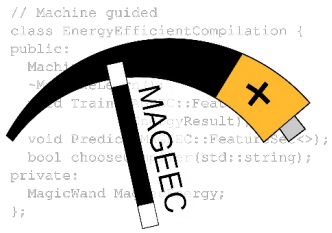
Energy  
measurement



Research into feedback  
directed optimization

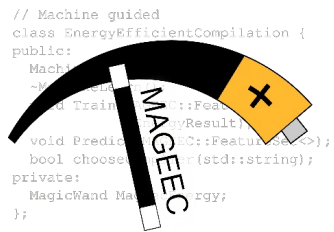


```
// Machine guided
class EnergyEfficientCompilation {
public:
    MagicWand Wand;
    ~MagicWand() {
        void Predict(MagicWand::FeatureSet<>);
        bool choose(MagicWand::FeatureSet<>);
    private:
        MagicWand MagicWand;
    };
};
```



# Literature Review

- Energy measuring and modeling
  - *The software drained my battery.* Kerstin Eder & Jeremy Bennett, NMI Yearbook 2012, [www.embecoscsm.com/resources/articles/#EAR12](http://www.embecoscsm.com/resources/articles/#EAR12).
- MILEPOST GCC - Feedback directed optimization
  - [ctuning.org/milepost-gcc](http://ctuning.org/milepost-gcc)
- Measurement of compiler energy usage
  - *Identifying Compiler Options to Minimize Energy Consumption for Embedded Platforms.* James Pallister, Simon Hollis, Jeremy Bennett [arxiv.org/abs/1303.6485](http://arxiv.org/abs/1303.6485)
- MAGEEC
  - [mageec.org](http://mageec.org)



# What's New?



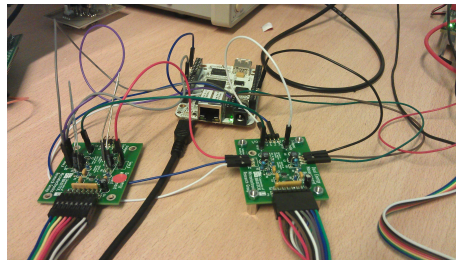
Objective is energy optimization

```
// Machine guided
class EnergyEfficientCompilation {
public:
    MagicWand Wand;
    ~MagicWand() { Wand.Destroy(); }
    void Predict(C::FeatureSet<> Result);
    bool choose(C::FeatureSet<> Result);
private:
    MagicWand Wand;
    MagicWand Wand;
};
```

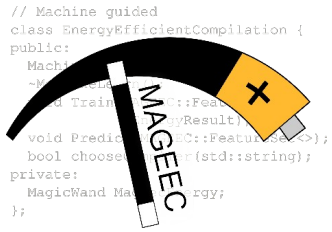
# What's New?



Objective is energy optimization



Energy measured *not* modeled



# Energy Measurement

- Based on ST Discovery board
  - ARM Cortex A8 with ADC daughter board
- Burst sample rate 6Msample/s
  - 192kB on-board RAM buffer
  - Short code samples e.g. superoptimizer
- Sustained sample rate 2Msample/s
  - Pre-processed and Streamed off-board
  - Used for MAGEEC
- The board in action:
  - [mageec.org/wiki/Power\\_Sensing\\_Board](http://mageec.org/wiki/Power_Sensing_Board)

```
// Machine guided
class EnergyEfficientCompilation {
public:
    Machine
    ~Machine() {}
    void Train(C::Feature, C::Result);
    void Predict(C::Feature, C::Result);
    bool choose(C::Result);
private:
    MagicWand MagicWand;
};
```

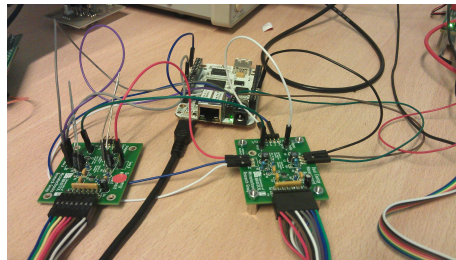
# What's New?



Objective is energy optimization



Generic framework: GCC *and* LLVM initially



Energy measured *not* modeled

```
// Machine guided
class EnergyEfficientCompilation {
public:
    MagicWand
    ~MagicWand() {}
    void Train(C::Feature, C::Result);
    void Predict(C::Feature);
    bool choose(C::Result);
private:
    MagicWand MagicWand;
};
```

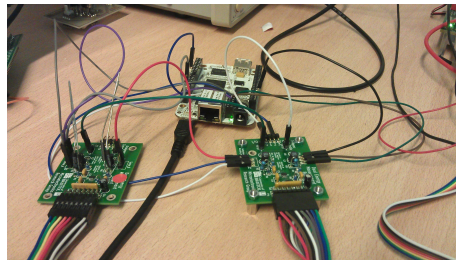
# What's New?



Objective is energy optimization



Generic framework: GCC *and* LLVM initially

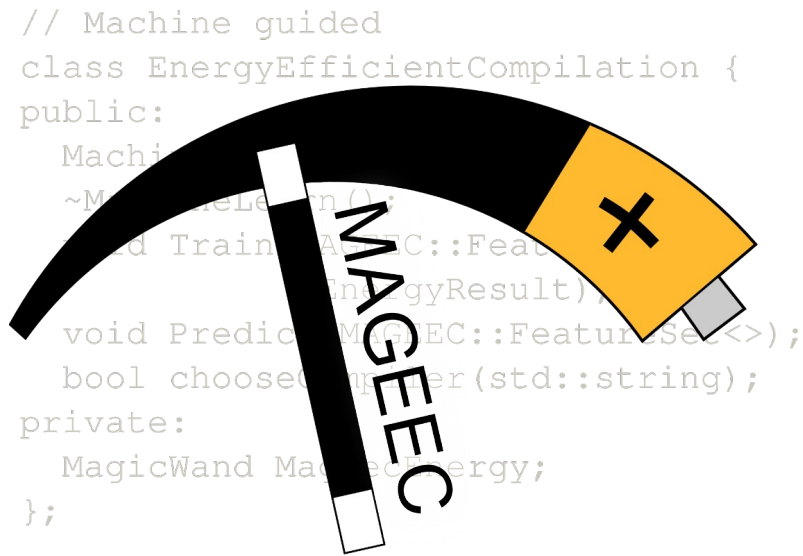


Energy measured *not* modeled

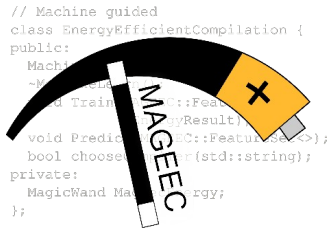


Working system, not research prototype



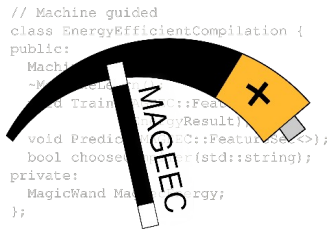


# Implementation

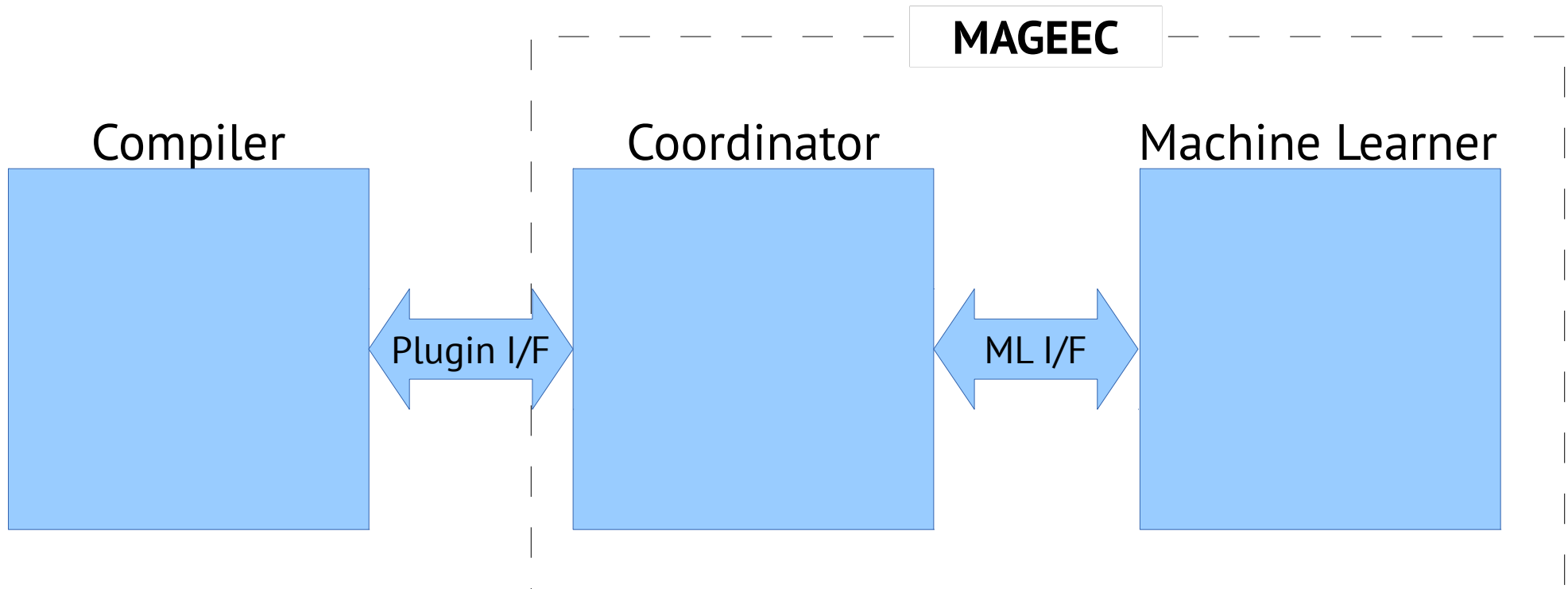


# Our Plan

- Implement MILEPOST concepts in a generic way.
- Train and evaluate based on real hardware energy measurements and existing passes.
- Write and evaluate optimization passes specifically for energy efficiency (Jörn Rennecke).



# Overall Design

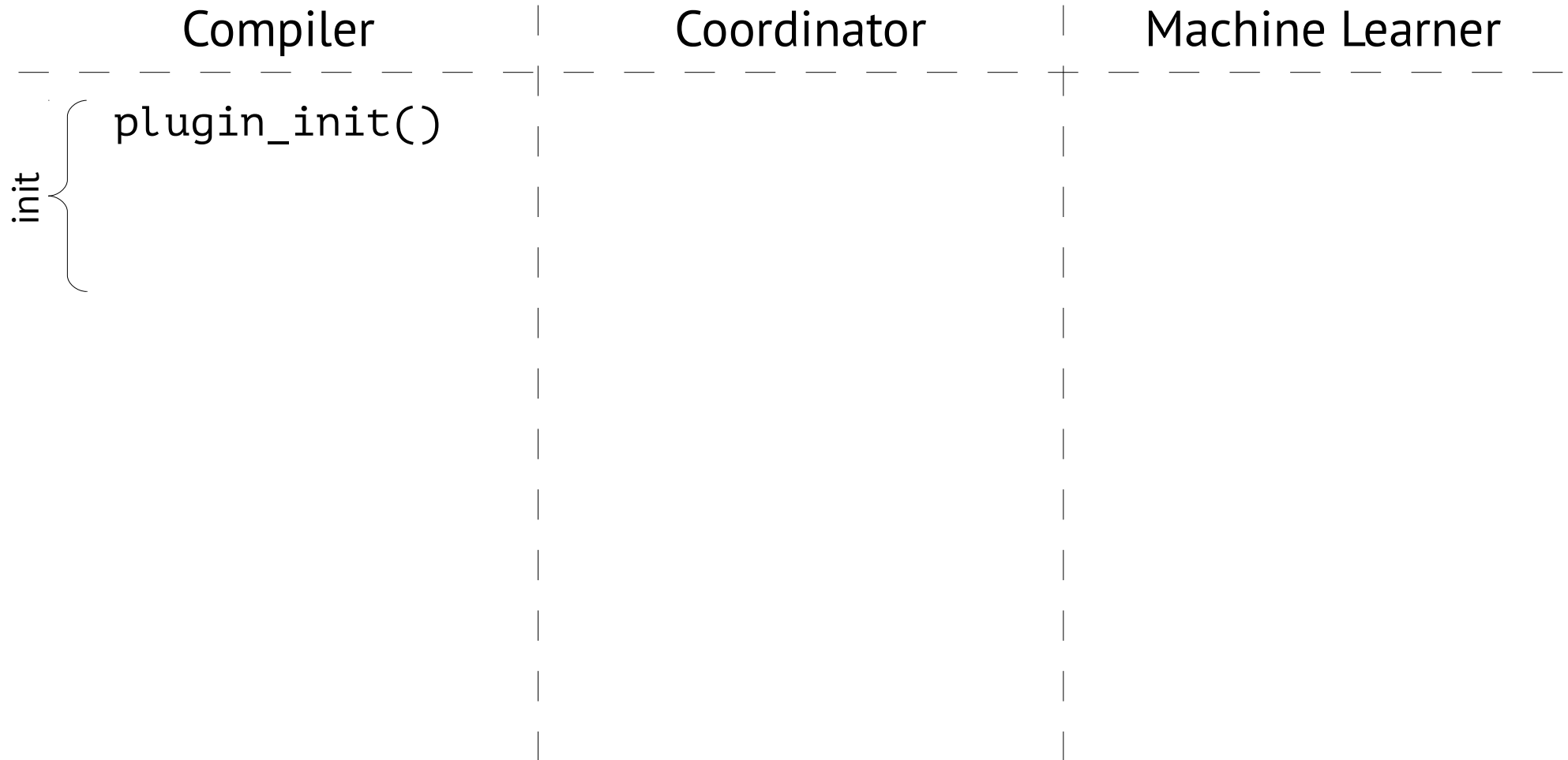


# Overall Design

[illegible]

\* may be `gen_features()`

# Overall Design

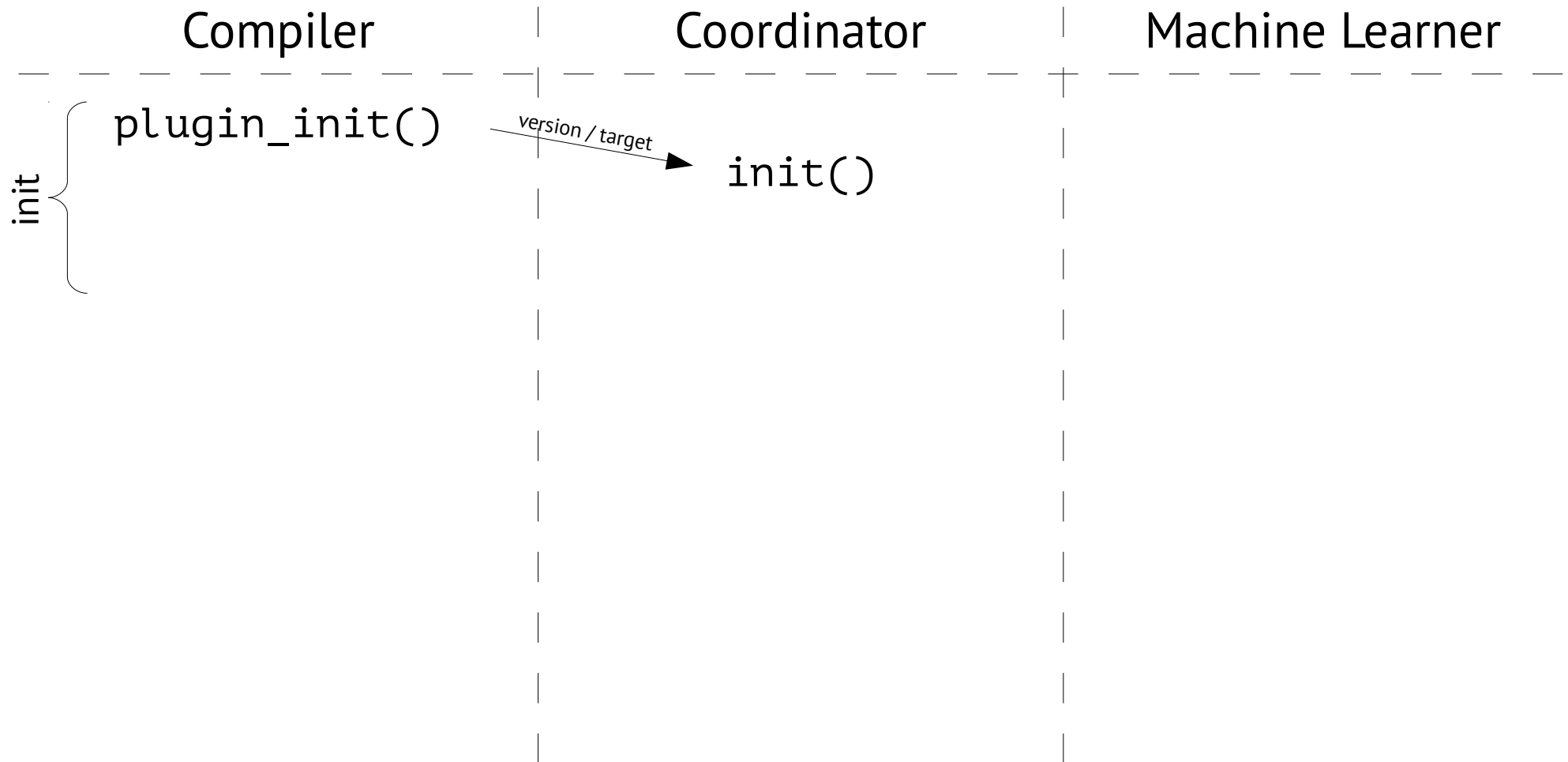


\* may be `gen_features()`

```
public:
    MagicWand() {}
    ~MagicWand() {}

    void Train(const std::vector<Feature>& features, const std::vector<Result>& results) {
        void Predict(const std::vector<Feature>& features) const {
            bool choose(const std::vector<Feature>& features) const {
        private:
            MagicWand MagicWand(const std::vector<Feature>& features);
    };
```

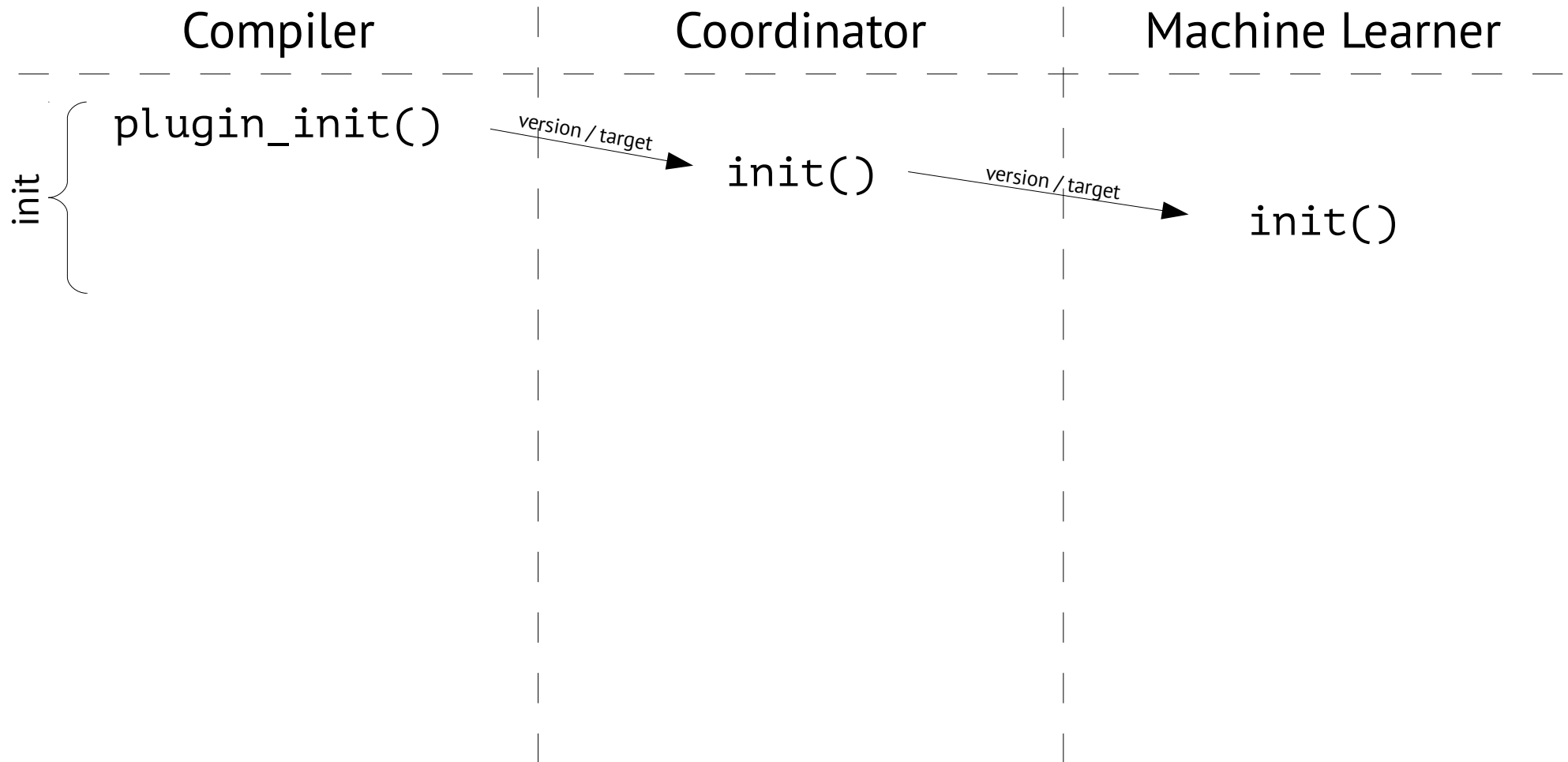
# Overall Design



\* may be `gen_features()`

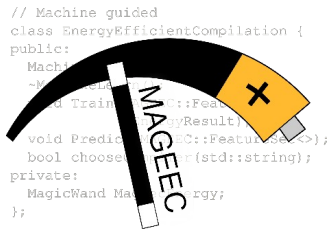
```
// Machine guided
class EnergyEfficientCompilation {
public:
    Machi
    ~Mach
    void Train(C::Feat
    void Predict(C::Feat
    bool choose(std::string);
private:
    MagicWand Ma
};
```

# Overall Design

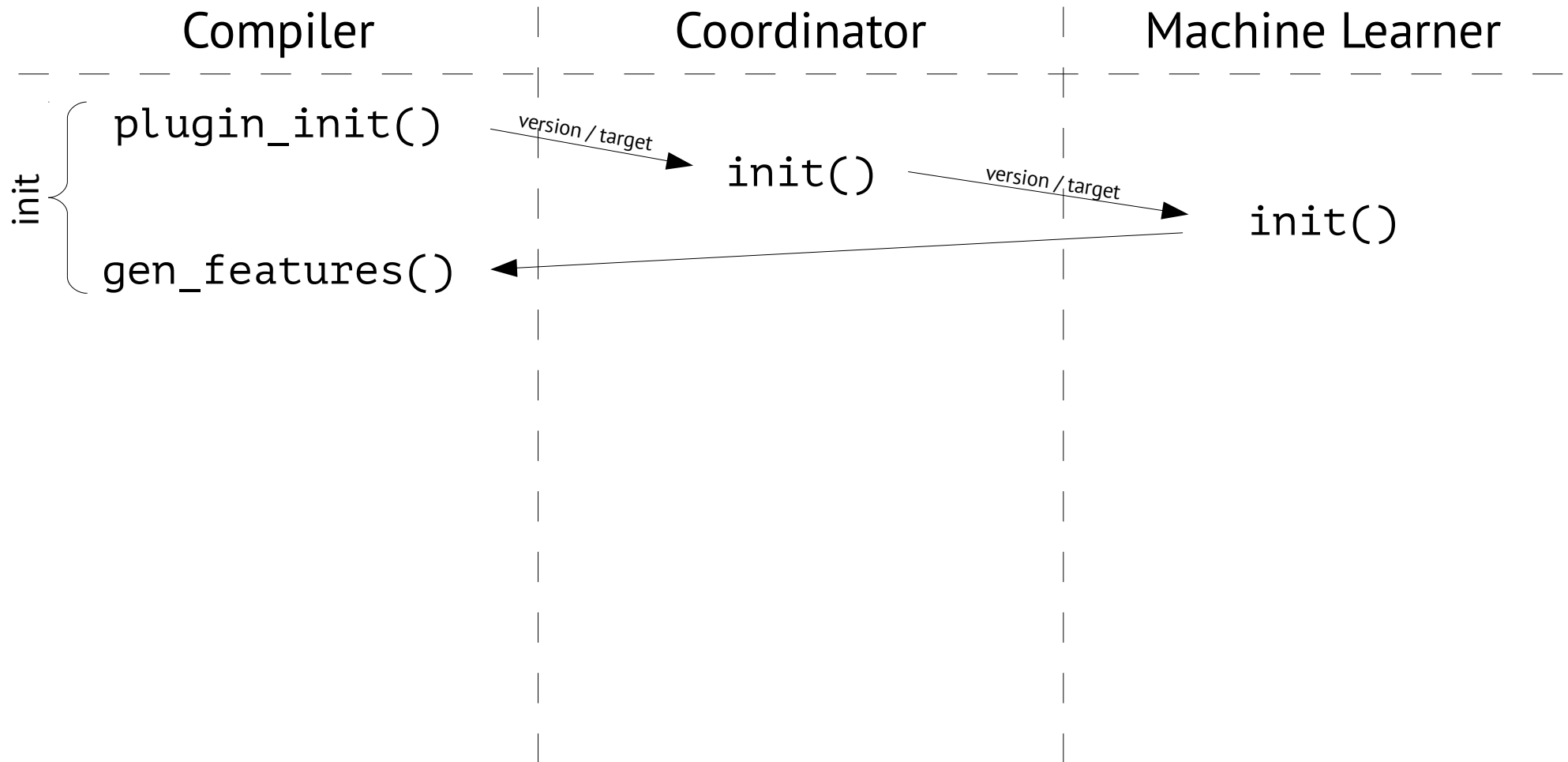


\* may be `gen_features()`

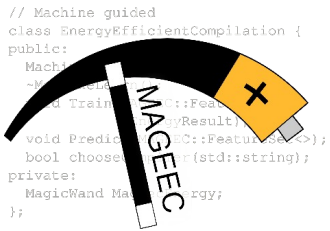




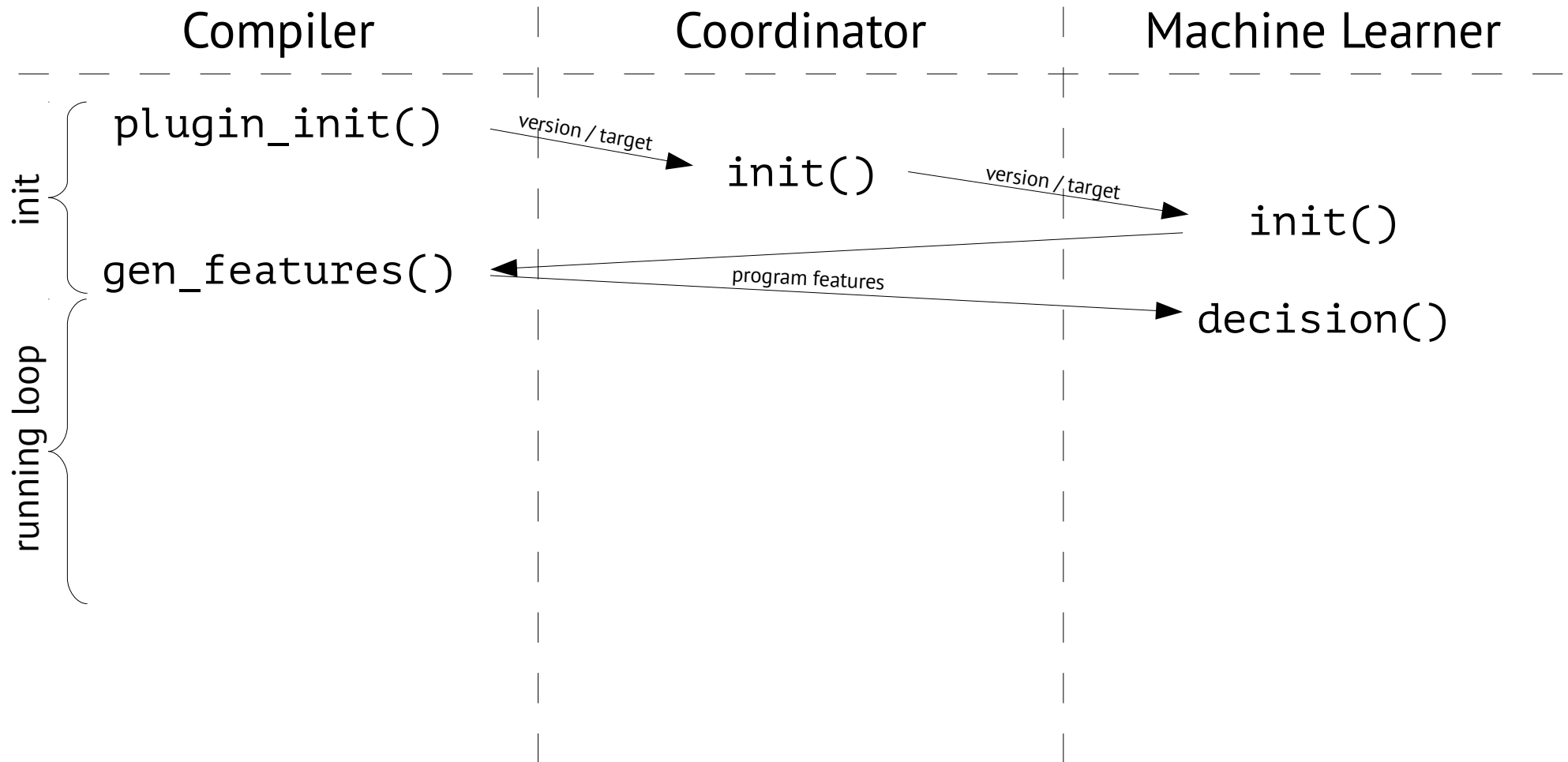
# Overall Design



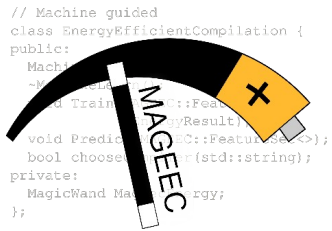
\* may be `gen_features()`



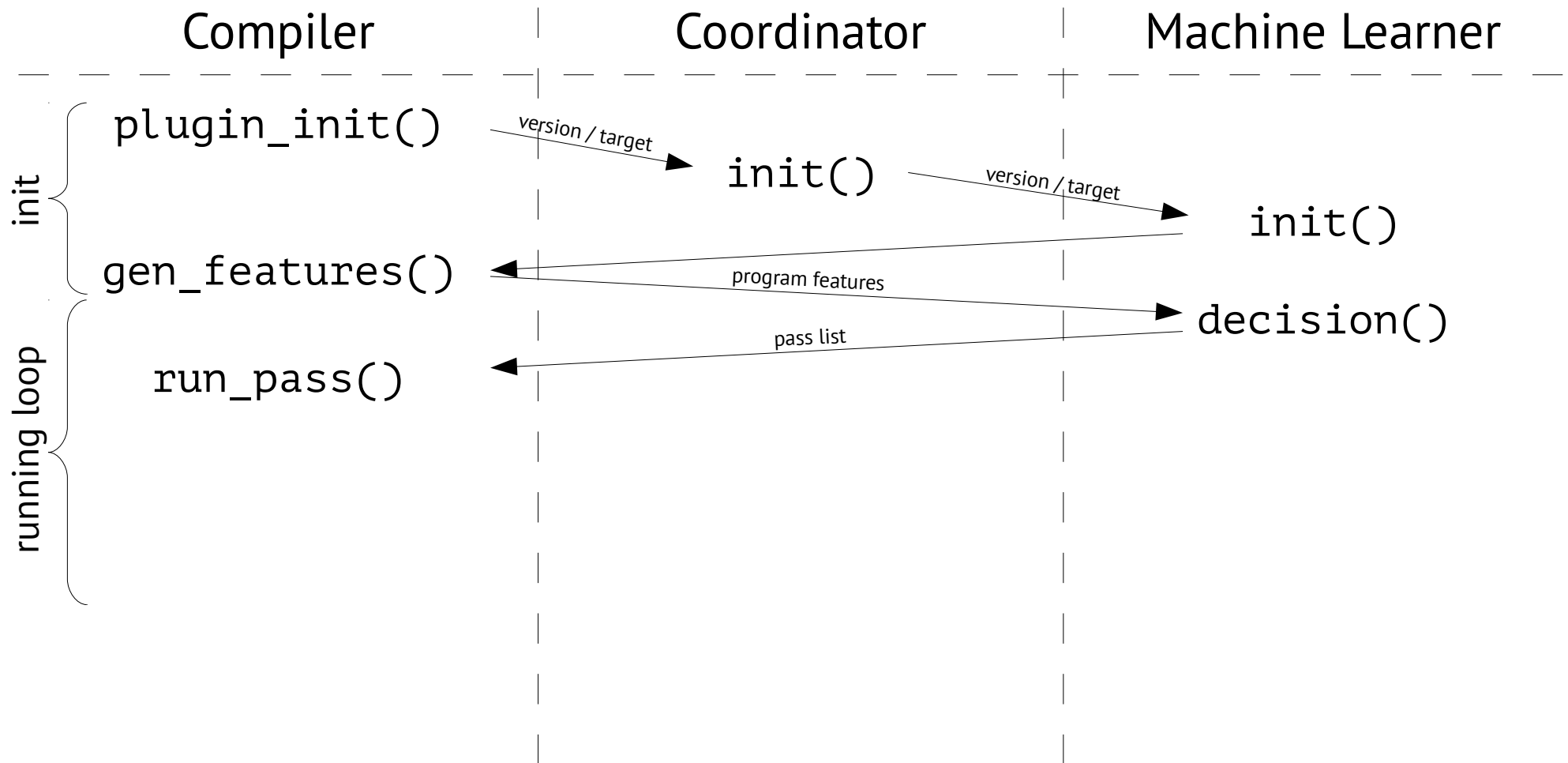
# Overall Design



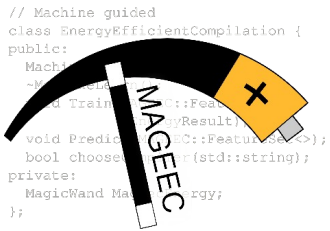
\* may be `gen_features()`



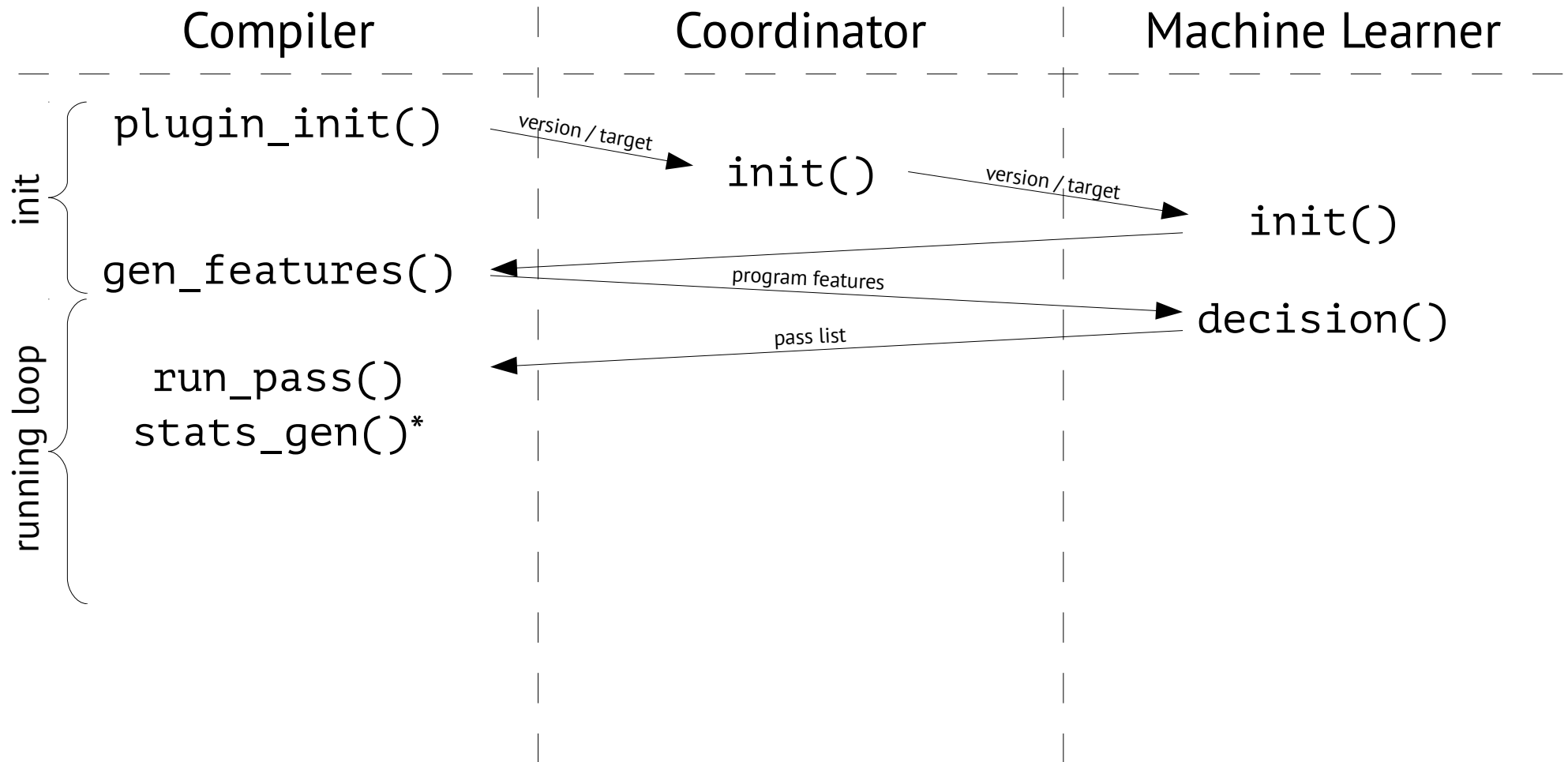
# Overall Design



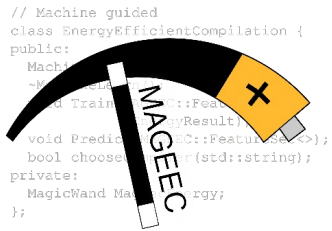
\* may be `gen_features()`



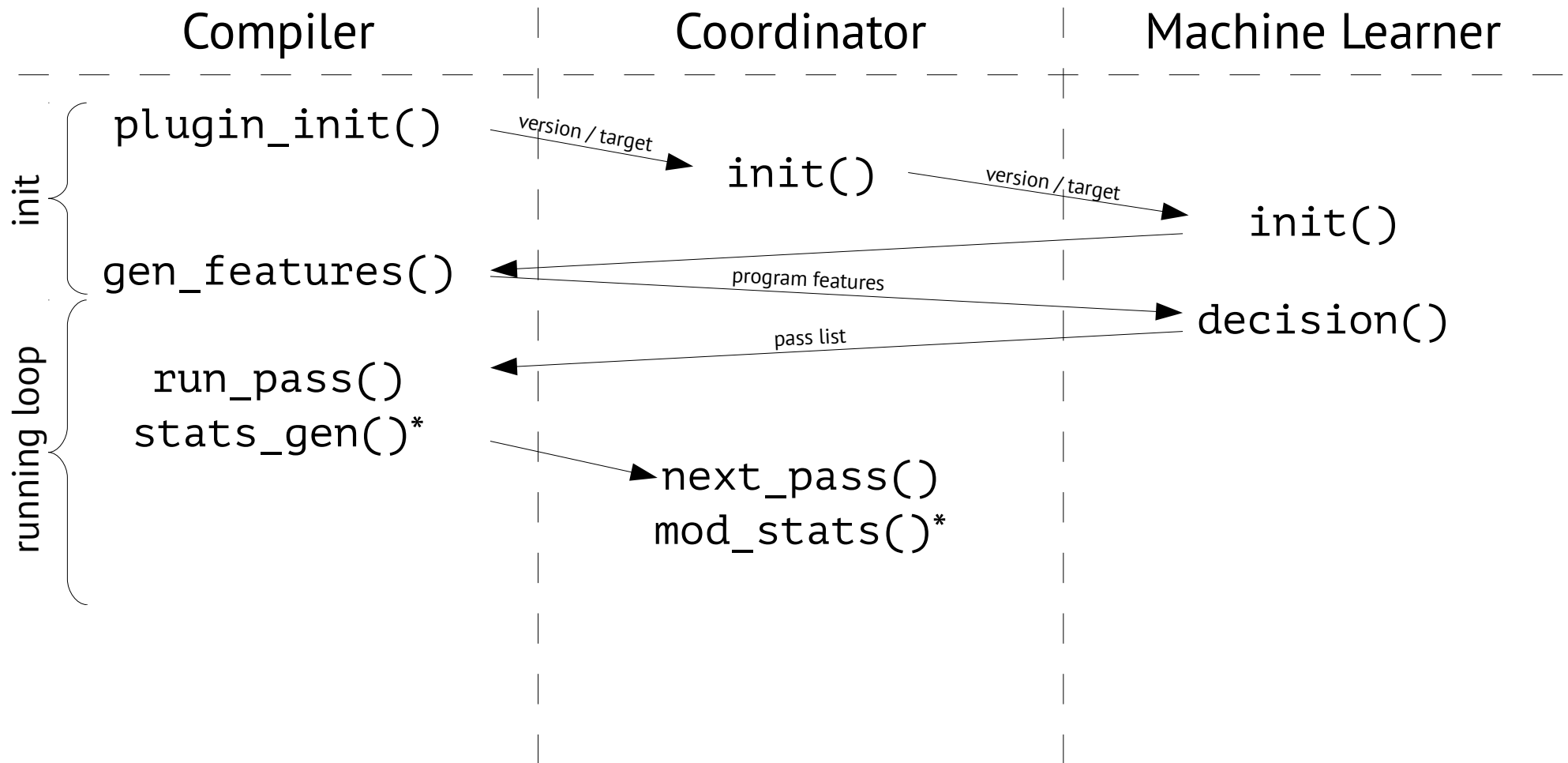
# Overall Design



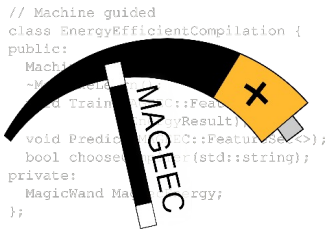
\* may be `gen_features()`



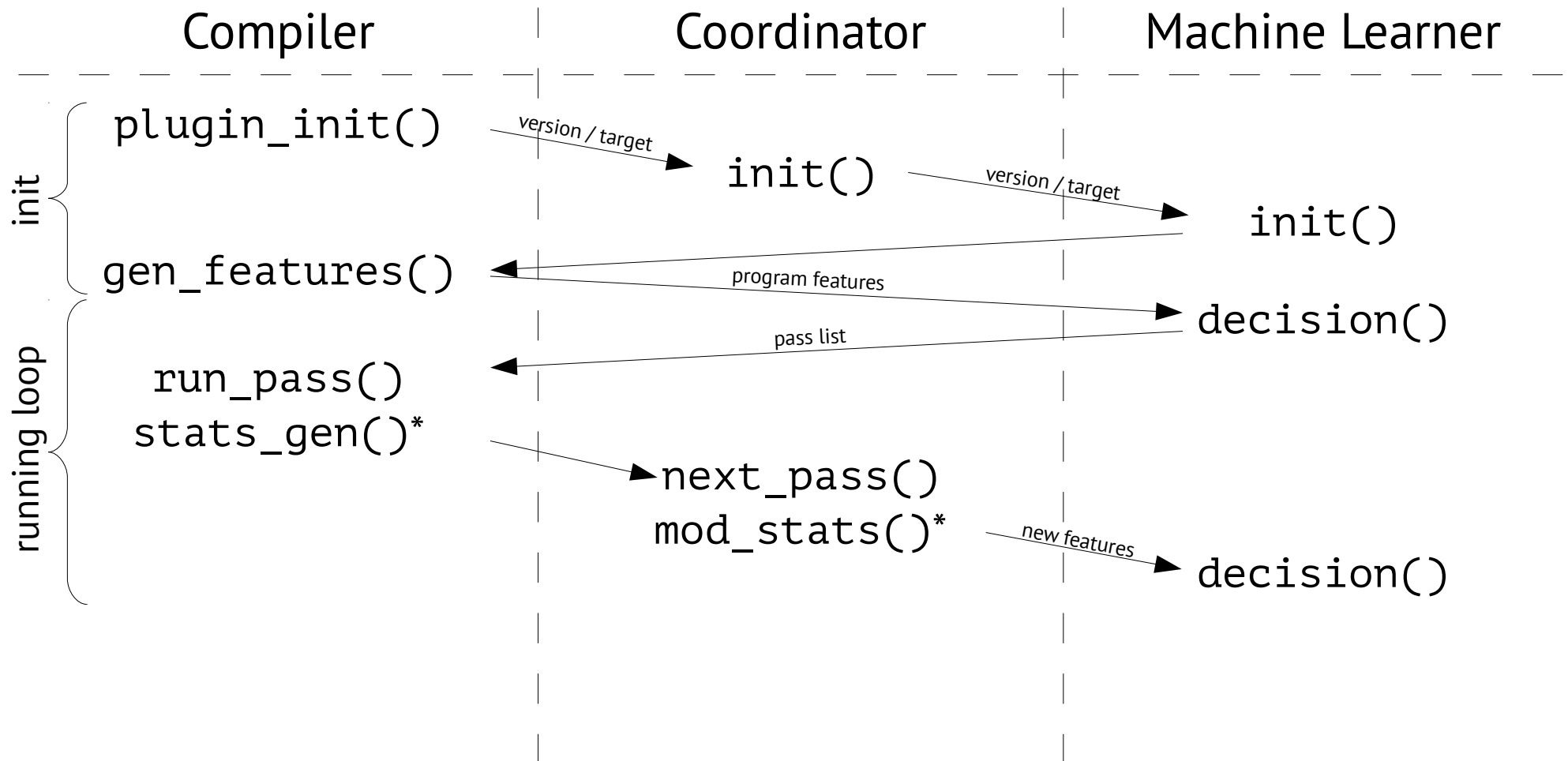
# Overall Design



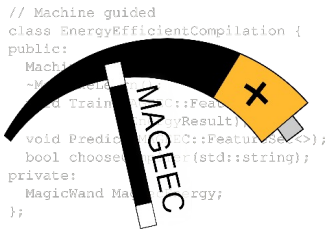
\* may be `gen_features()`



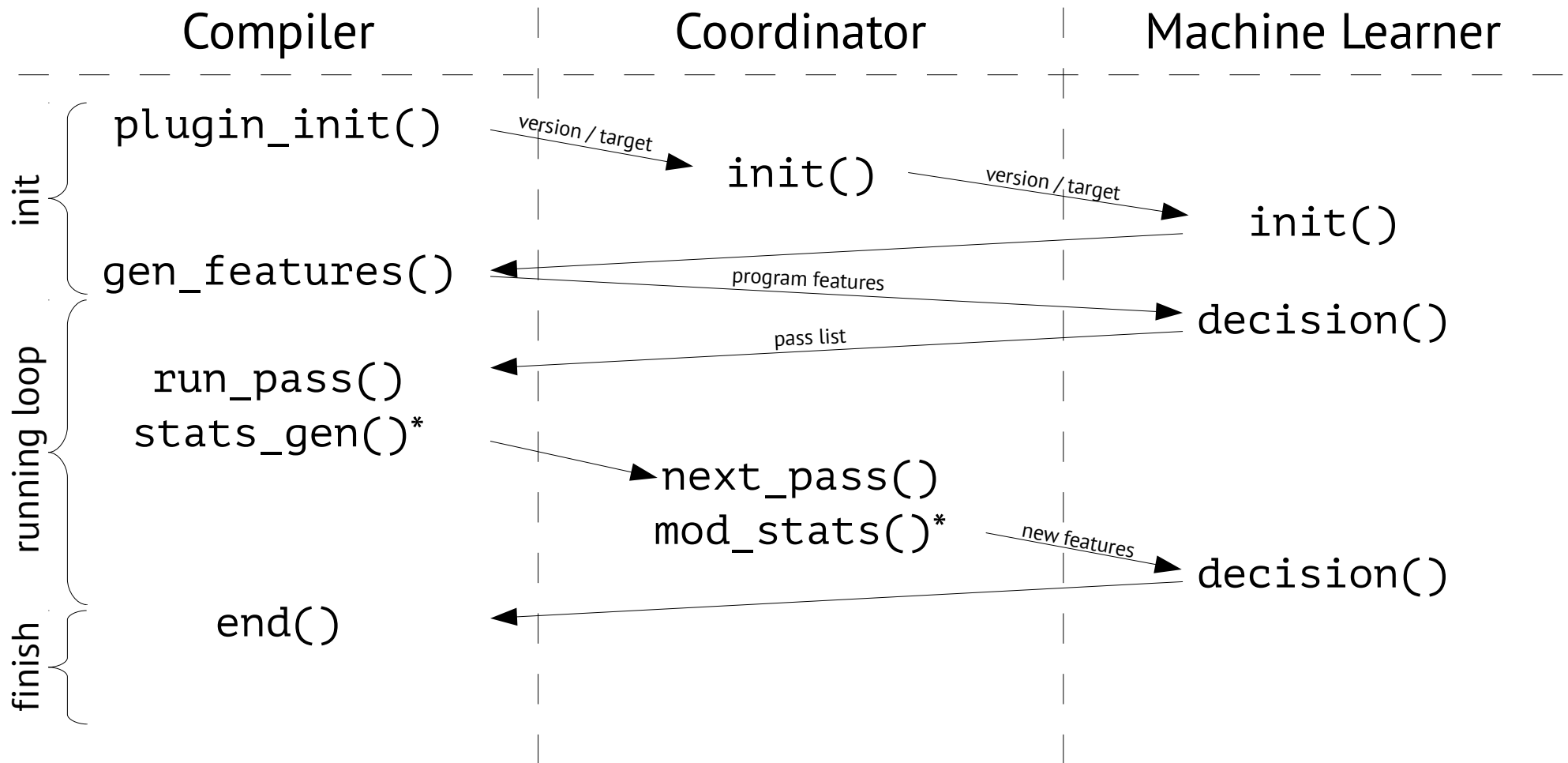
# Overall Design



\* may be `gen_features()`

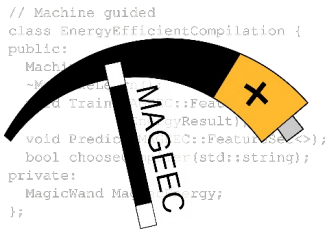


# Overall Design

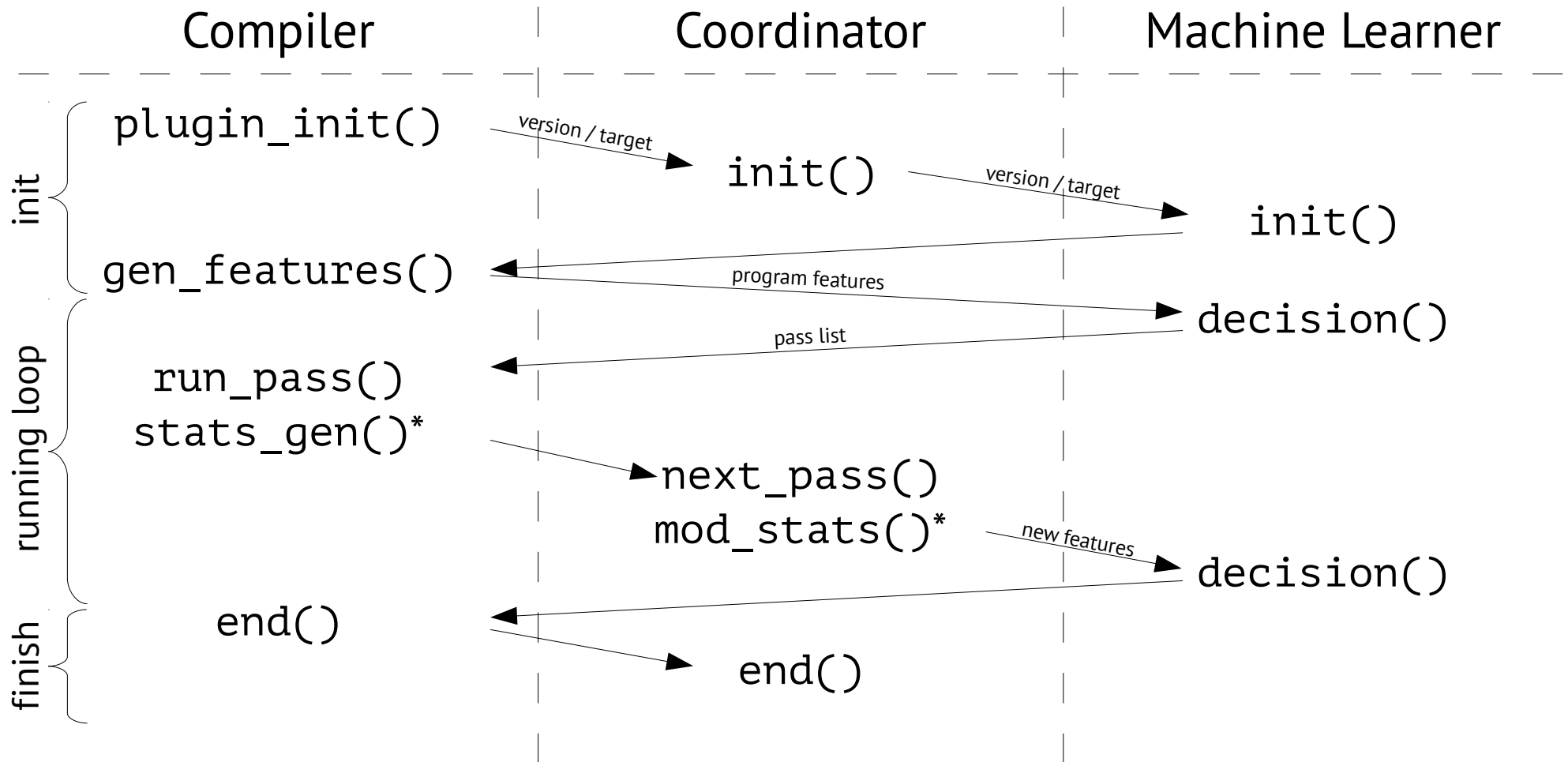


\* may be gen\_features()

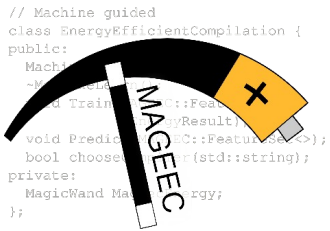




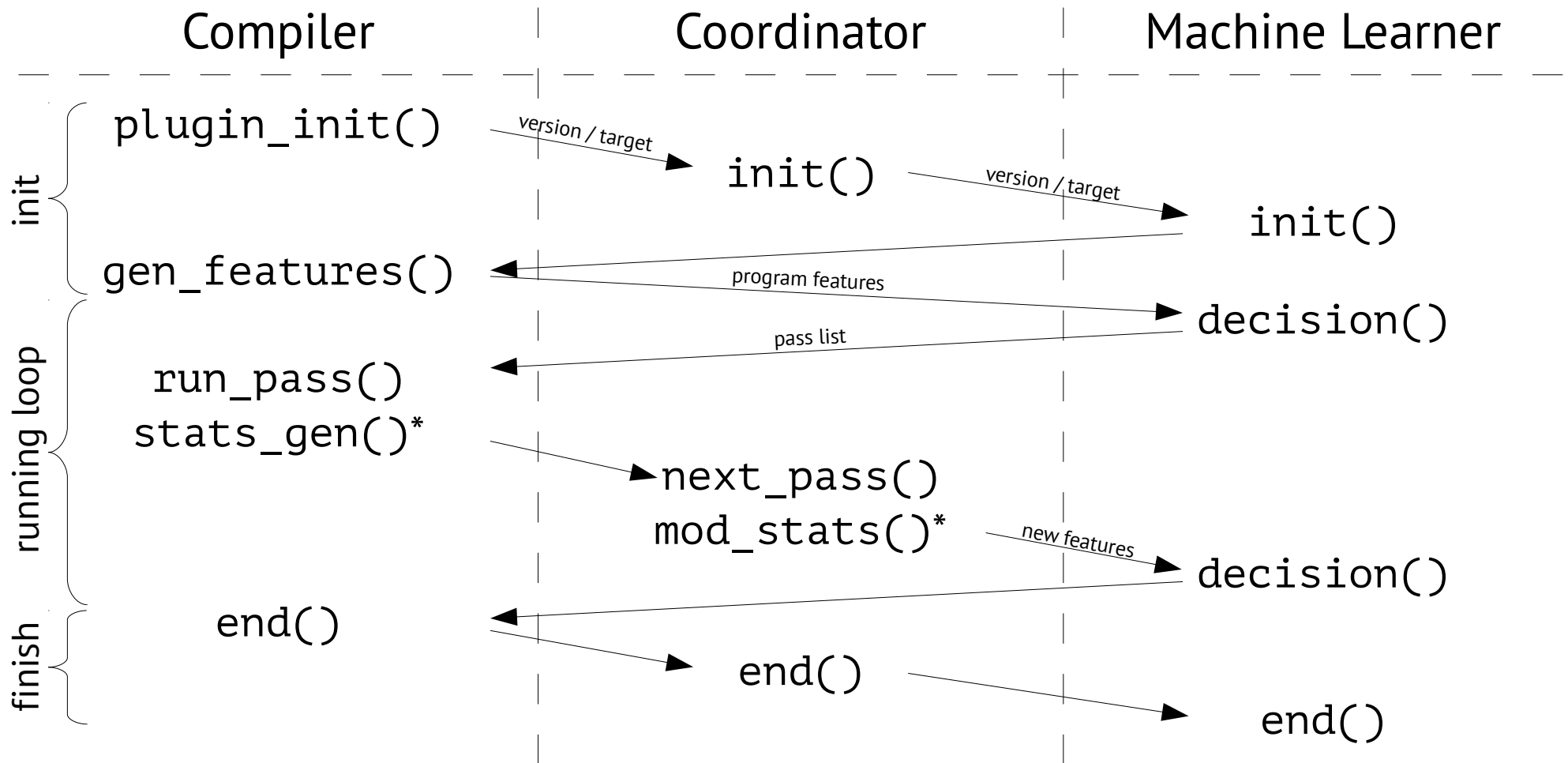
# Overall Design



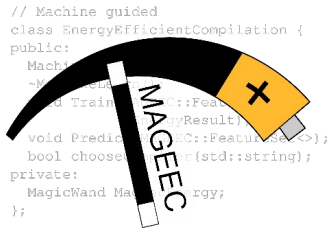
\* may be gen\_features()



# Overall Design

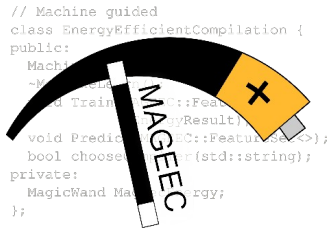


\* may be gen\_features()



# Pass Constraints

- *The challenge for an arbitrary order pass manager: **only using valid combinations.***
- The machine learner can additionally learn about what options to avoid when selecting passes.



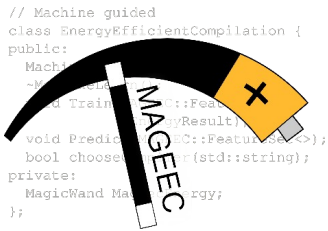
# Flags

- Today's optimisation passes optimize for speed... again, why not have passes ***dedicated*** to energy?

## gcc -fenergy

(or a more descriptive name)

# Community Involvement



**Category:Planning - MAGEEC**  
MAchine Guided Energy Efficient Compilation

Home Planning Design

**Category:Planning**  
This category holds all documents related to the planning stage of the MAGEEC project.

A PDF version of core documents is available here.

Pages in category 'Planning'

The following 4 pages are in this category:

- [Milestones](#)
- [Planning](#)
- [Project Plan](#)
- [Risk Register](#)

**Category:Design - MAGEEC**  
MAchine Guided Energy Efficient Compilation

Home Planning Design

**Category:Design**  
This page contains all the design related documents of the MAGEEC project.

Pages in category 'Design'

The following 4 pages are in this category:

- [Design](#)
- [Interface Flow](#)
- [Literature](#)
- [Power Sensing Board](#)

**Interface Flow - MAGEEC**  
MAchine Guided Energy Efficient Compilation

Home Planning Design

**Interface Flow**  
This page describes the interfaces and basic flow of how the three components of MAGEEC will communicate. Between the components are the direction of communication along with what data is communicated.

Compiler		MAGEEC		ML	Comment
plugin_init	->				
	v/target				
		init	->		
				init	
			<-		
		needpass?			(only if needed)
genpasslist	<-				
	->				
		build/load list	->	list + vtarget	
				load_db	(based of v/target)
			<-		
gen_features	<-				
	->				(Need to define features for here + what to do if not avail)
	src/name				
			->	features	
				decision	(Initial decision, pass list of N passes for each pass)

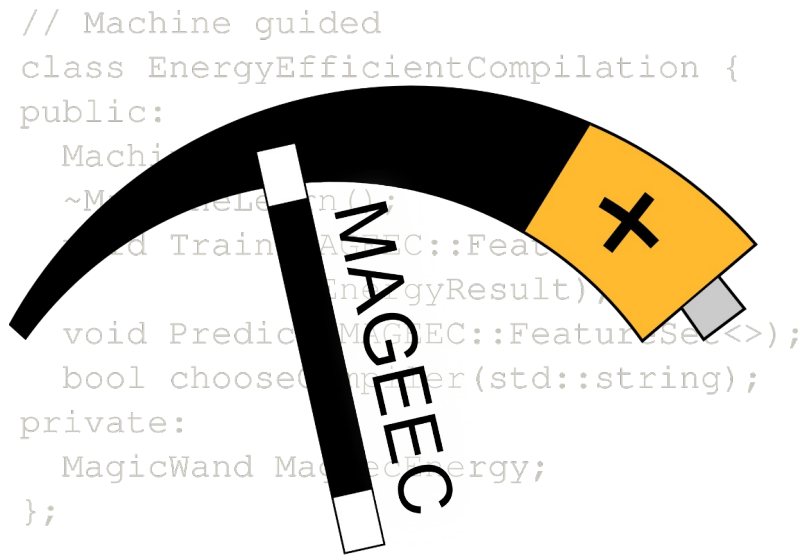
**VIEWS**

- [View](#)
- [Edit](#)
- [History](#)
- [Delete](#)
- [Move](#)
- [Protect](#)
- [Unwatch](#)

**WIKI SEARCH**

Search

**TOOLBOX**



# Thank you

[mageec.org](http://mageec.org)

[www.embecoscsm.com](http://www.embecoscsm.com)

[cs.bris.ac.uk](http://cs.bris.ac.uk)