# BRINGING EXPERT SYSTEM INTO RUBY

**Jaroslav Čecho**
Bachelor Degree Programme (1), FIT BUT
E-mail: xcecho00@stud.fit.vutbr.cz

Supervised by: Zdeněk Letko
E-mail: iletko@fit.vutbr.cz

## ABSTRACT

This paper describes a library called rbClips that makes CLIPS functionality available from Ruby. CLIPS is a public domain tool for building expert systems that was originally developed in NASA in 1990's. The tool itself is written in C but its user interface is very similar to Lisp language. Ruby is a modern dynamic scripting language that offers programmer flexible syntax, purely object environment, openness of objects and other interesting features. The library is ment to be used to build expert system for detection of possibly malicious code in antivirus software.

## 1 ÚVOD

V tomto článku bych rád popsal svou bakalářskou práci, ve které jsem propojil dvě technologie, systém pro tvorbu expertních systémů CLIPS a mladý skriptovací jazyk Ruby, do jednoho funkčního celku.

**Ruby**. Ruby [1] je mladý dynamický skriptovací jazyk vytvořený japonským inženýrem Yukihiro Matsumoto známým na internetu pod přezdívkou Matz. Na popularitě získává až v poslední době kvůli dřívější absenci anglicky psaných materiálů. Nabízí programátorovi plně objektový jazyk s volnou syntaxí (například možnost vynechat závorky okolo argumentů funkcí na místech, kde to není syntakticky nejednoznačné). Podobně jako například prototypový Javascript umožnuje jakoukoliv třídu znovu otevřít a upravit chování jeho metody nebo dokonce celou metodu nově vytvořit. Navíc lze takovéto otevření provádět i u jednolivých objektů.

**Expertní systémy.** Expertní systém [2] je program, který umí odpovídat na dotazy uživatele na základě uložených znalostí. Svou funkčností simuluje rozhodování lidského experta a používá se v případech, ve kterých neexistuje algoritmické řešení, případně na místech kde by takové řešení bylo přílis složité či špatně udržovatelné. Zpravidla se skládá z několika částí: (1) Báze znalostí, kde jsou uložena pravidla umožnující měnit, mazat a vyvozovat nová fakta z pracovní paměti, (2) pracovní pamět uchovavající známá fakta, (3) řídící algoritmus vykonavající vyhovující pravidla, (4) vysvětlující podsystém pro zdůvodnění dosaženého výsledku a také (5) uživatelské rozhraní pro komunikaci s uživatelem (ať už formou příkazového řádku nebo grafického rozhraní).

CLIPS je systém pro tvorbu expertních systémů, dnes dostupný jako public domain software. Je stále udržován původním autorem a mimo této verze existují i další jeho varianty, které původní

CLIPS obohacují o nové možnosti - například FuzzyClips, což je rozšíření o možnosti pracovat s fuzzy logikou nebo přepis pro Javu jménem Jess. Celý systém je napsaný v dobře přenositelném jazyce C, ovšem jazyk pro ovládání systému je velice podobný jazyku LISP (viz. obrázek 1).

```
(assert (zvire (jmeno "Azor") (vek 3) (rasa "vlcak")))
```

**Obrázek 1:** Ukázka vytvoření faktu v CLIPS.

Existují dva různé způsoby zpřístupnění funkcionality knihoven psaných v jazyku C do skriptovacích jazyků – manuální a automatický. V prvním případě si programátor napíše rozhraní mezi knihovnou a jazykem zcela sám a má tedy plnou kontrolu nad podobou výsledku. V druhém případě lze využít utility, které toto rozhraní umí vygenerovat automaticky samy. Příkladem může být projekt SWIG, který zpracuje zdrojové kódy knihovny psané v C (případně C++), pro které vygeneruje obalovací kód tak, aby šly přeložit a použit jako binarní rozšíření zvoleného cílového skriptovacího jazyka. V případě SWIG patří mezi podporované cílové jazyky Perl, PHP, Python, Tcl či právě Ruby. Tento způsob je velice rychlý a přímočarý, bohužel v tomto případě má programátor minimální možnost ovlivnit výsledek.

## 2 KNIHOVNA RBCLIPS

V rámci své bakalářské práce jsem vytvořil knihovnu rbClips, kde jsem manuálně zpřístupnil funkce CLIPS z Ruby. Manuální přístup převodu mi předně umožnil zapouzdřit procedurální chování CLIPS do objektů i přizpůsobit rozhraní zvyklostem objektového návrhu a nepsaným hojně rozšířeným konvencím Ruby. Aplikační rozhraní (API) jsem navrhl po vzoru projektu ActiveRecords (AR). AR je knihovna zapouzdřující práci s relačními databázemi do objeků. Původne vznikla pro potřeby webového frameworku Ruby On Rails, ale lze ji používat zcela nezávisle. Programátor se základní znalostí AR se velice rychle zorientuje i v používání rbClips.

Navíc jsem tak získal možnost odstínit koncového uživatele od znalosti syntaxe uživatelského rozhraní CLIPS. Některé méně využívané konstrukce nejsou bohužel v rbClips podporovány, a proto jsem přidal možnost vkládat a vykonávat validní úryvky CLIPS kódu přímo. Možnost zapouzdřit CLIPS a odebrat tak pro uživatele nutnost znalosti syntaxe velice podobné jazyku Lisp byl jeden z hlavních důvodu proč jsem se rozhodl pro manuální přístup.

Dalším důležitým důvodem je umožnění zpětného volání Ruby metod. Ve třídě pro práci s pravidly lze jako akci (konsekvent) nastavit libovolný Ruby objekt a jeho metodu, jež po aktivaci pravidla může být zavolána i s aktivačními parametry. Pro tuto funkcionalitu jsem si musel v CLIPS vytvořit vlastní funkci, která je schopna zavolat na libovolném registrovaném Ruby objektu definovanou metodu. Samotná registrace je pro uživatele plně transparentní a nemusí se jí zabývat. Díky tomu není v rámci pravidel uživatel svázán pouze schopnostmi stabilního jádra CLIPS, ale otevírají se mu úplně nové možnosti. Do výsledného expertního systému může integrovat libovolnu knihovnu Ruby a plně využít všech možností, které tento jazyk nabízí.

Z technického pohledu je rbClips knihovna obsahující modul `Clips` se čtyřmi zakladními a několika podpůrnými třídami. Základní třídy zapouzdřují možnost ovládání prostředí CLIPS (třída `Base`), prací se šablonami (třída `Template`), fakty (`Fact`) a nakonec také správu pravidel (`Rule`). Podpůrné třídy slouží například pro manipulaci s prostředími (`Environment`) či poskytují možnost ovlivnit přípustné hodnoty v jednotlivých pojmenovaných slotech faktů (`Constraint`).

## 3 AKTUÁLNÍ STAV

Knihovna ještě není zcela dokončena. Funguje veškerá popsaná funkcionalita až na umožnění zpětného volání Ruby objektů, které je napsáno jen z části a není plně stabilní.

## 4 REÁLNÉ NASAZENÍ

Tento projekt vznikl v rámci společnosti AVG Technologies s.r.o s cílem napodobit rozhodovací proces lidských odborníků v oblasti analýzy virových vzorků. Knihovna bude využita pro automatické napodobení postupu jejich analýzy, kterou by nad daným vzorkem prováděli. Bude rozhodovat zda-li je vzorek malware nebo čistý legitimní software. Výhody řešení postaveného pomocí rbClips jsou (1) jednoduché propojení programu s relačními databázemi, kde jsou uloženy různé záznamy či hledané kontrolní součty, (2) snadnost volání externích utitil a rozhodování na základě jejich výsledku po vzoru lidského pracovníka a (3) možnost automaticky vytvářet nová pravidla podle zjištěných skutečností, tedy strojové učení.

Další využití by knihovna mohla najít na straně klienta AVG, tedy u uživatelů antivirového produktu, kde by expertní systém posloužil jako základ pro detekci malware na základě chování (behaviour detection). Operační systém bude hlásit antivirovému programu jednotlivé akce, které programy postupně provádějí (například alokace velkého bloku souvislé paměti, rozšifrování dat do tohoto bloku a jeho následné spuštění). Ten by rozhodoval na základě výstupu z rbClips umožnil varovat uživatele před podezřelým chováním jednolivých programů.

## PODĚKOVÁNÍ

Rád bych poděkoval technickému vedoucímu této prace, Ryanu Hicksovi, za nápad, rady a ochotu řešit problémy, na které jsem při tvorbě narazil a svému odbornému vedoucímu Zdeňkovi Letkovi za jeho trpělivost a čas věnovaný kontrole korektnosti mých textů.

## REFERENCE

[1] David Flanagan, Yukihiro Matsumoto: The Ruby Programming Languag, O'Reilly Media, Inc 2008, ISBN 0-596-51617-7

[2] Josph C. Giarratano, Gary D. Rile: Expert Systems: Principles and programming, Thomson Learning, Inc 2005, ISBN 0-534-38447-1