Michał **BUGAŁA**

HYDROGEN – ŚRODOWISKO PROGRAMISTYCZNE DLA SYSTEMU VIRTUAL BATTLESPACE

Streszczenie. W artykule przedstawiono możliwości i rozwiązania technologiczne środowiska programistycznego Hydrogen, powstałego w Biurze Symulatorów OBRUM sp. z o.o. i przeznaczonego dla wirtualnego systemu pola walki Virtual Battlespace 3. Hydrogen jest uzupełnieniem narzędzi dostarczanych przez producenta systemu VBS, firmy Bohemia Interactive, i w łatwy sposób umożliwia rozbudowę systemu o nowe funkcjonalności.

Słowa kluczowe: Virtual Battlespace, środowisko programistyczne, programowanie, język skryptowy, edytor interfejsu graficznego.

1. WPROWADZENIE

Wielu doświadczonych programistów porównuje programowanie do sztuki i nie jest przypadkiem, że każdy z nich wyróżnia się niezwykłą estetyką zapisu własnych programów komputerowych, potocznie zwanych kodem źródłowym. Tylko dobrze i czytelnie napisany kod instruktażowy może stać się natchnieniem do zagłębienia się w tematykę danego języka programowania oraz twórczą inspiracją do rozwijania własnego kodu. Głównym instrumentem, jakim posługuje się programista, jest środowisko programistyczne, czyli aplikacja, która umożliwia edycję kodu źródłowego oraz ułatwia sfinalizowanie procesu twórczego (kompilację, asemblację itp.).

Artykuł prezentuje możliwości oraz rozwiązania technologiczne zastosowane w aplikacji Hydrogen powstałej z myślą o szybszym, sprawniejszym i bardziej eleganckim przygotowywaniu skryptów i plików konfiguracyjnych w systemie Virtual Battlespace (VBS) [1]. System VBS jest szeroko stosowanym rozwiązaniem militarnym, wykorzystywanym przez większość centrów symulacji państw członkowskich NATO i służy do przeprowadzania różnych scenariuszy wirtualnego pola walki oraz szkoleń taktycznych [2]. Skrypty i pliki konfiguracyjne służą rozbudowie systemu VBS o nowe funkcjonalności; umożliwiają dodanie pojazdów, broni czy statków powietrznych, które nie występują w systemie.

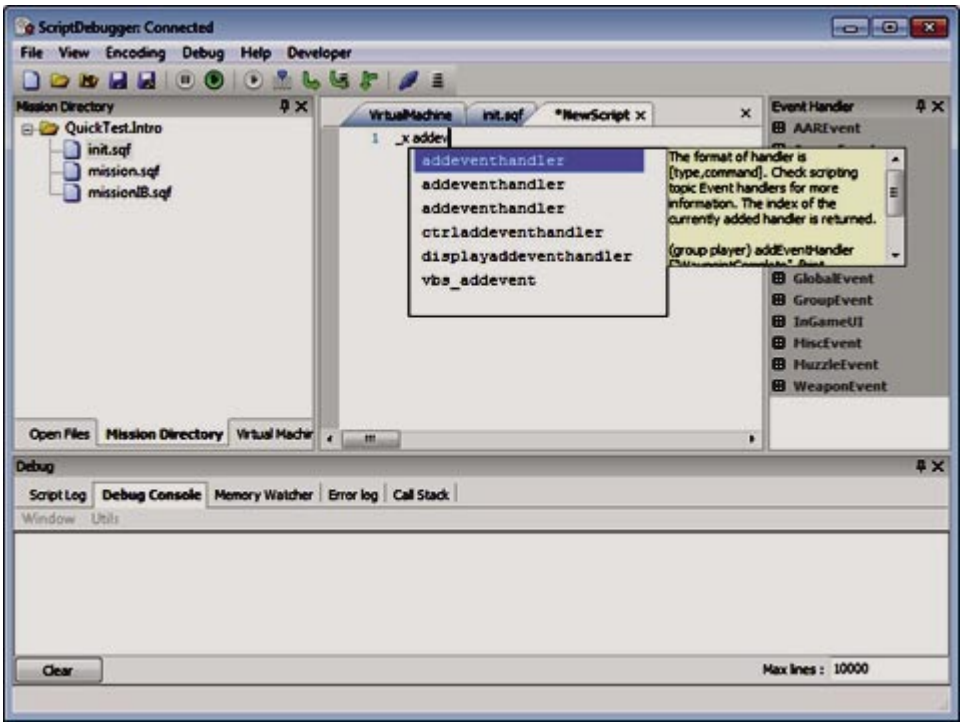
Praktyczne wykorzystanie środowiska programistycznego Hydrogen pozwoliło wyeliminować potrzebę używania bardzo ograniczonego narzędzia dostarczanego z deweloperską wersją systemu VBS o nazwie Script Debugger Plugin (rys. 1) [3]. Script Debugger Plugin systemu VBS posiada system podpowiedzi, jednak jest on ograniczony jedynie do zdarzeń (z ang. *events*) języka skryptowego SQF, a możliwość debugowania[1] - tylko do skryptów SQF uruchamianych w misji[2].

[1] Interpretacja kodu źródłowego w celu wykrycia błędów[4].

[2] Oprócz skryptów SQF, system wymaga dodania szeregu plików konfiguracyjnych, które nie są debugowane w programie Script Debugger Plugin. Debugowanie skryptów SQF po uruchomieniu misji (symulacji) wymaga przeprowadzenia wielu dodatkowych czynności, które bardzo utrudniają zasadnicze prace programistyczne.

mgr inż. Michał **BUGAŁA** – Ośrodek Badawczo-Rozwojowy Urządzeń Mechanicznych „OBRUM” sp. z o.o., Gliwice

Środowisko programistyczne Hydrogen zostało wykonane przez autora artykułu w wyniku braku podobnych rozwiązań na rynku i potrzeby użycia sprawnego narzędzia, umożliwiającego przygotowanie systemu szkoleniowego dla wojsk inżynieryjnych przez Biuro Symulatorów OBRUM sp. z o.o., realizowanego na zlecenie Wojskowej Akademii Technicznej.



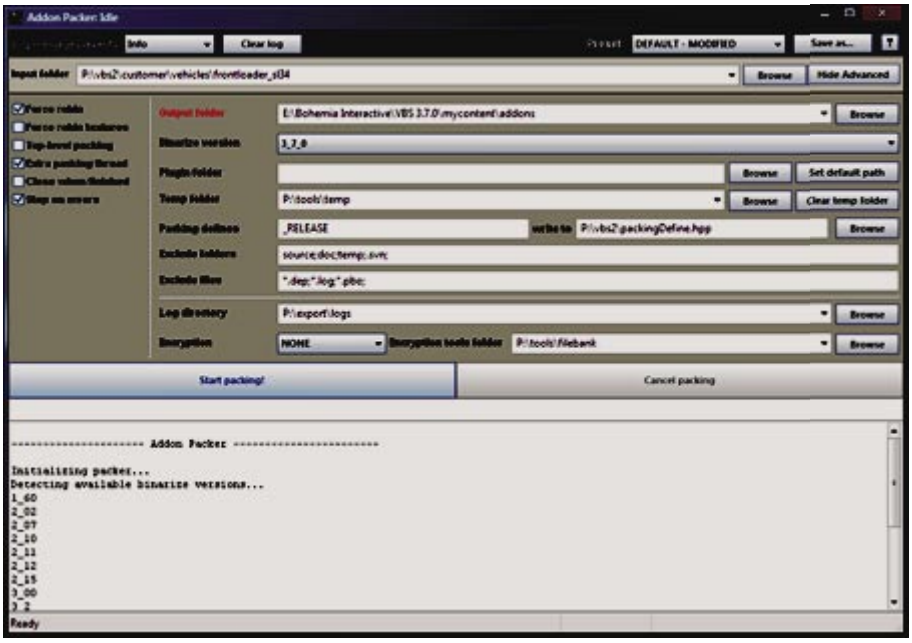
Rys. 1. Ekran główny standardowego środowiska programistycznego systemu VBS [3]

2. IMPLEMENTACJA ŚRODOWISKA HYDROGEN

Hydrogen został napisany w nowoczesnym dialekcie języka Pascal (kompilator Free Pascal) [5] przy użyciu środowiska programistycznego o nazwie Lazarus [6]. Kompilator oraz środowisko programistyczne Lazarus rozpowszechniane są na zasadach wolnego oprogramowania (otwarty kod źródłowy), a ich architektura stała się inspiracją do napisania środowiska programistycznego Hydrogen. Hydrogen posiada wszystkie funkcjonalności, jakie powinien zawierać dobry edytor tekstu lub edytor kodu źródłowego (pozycja kursora, informacje o zmianie w tekście, lista ostatnio otwartych plików itp.). Ponadto zaimplementowano graficzny edytor interfejsu użytkownika, rozbudowany system pomocy oraz system przeglądania klas systemu VBS. Lista głównych cech opracowanych narzędzi środowiska Hydrogen została zamieszczona w Dodatku.

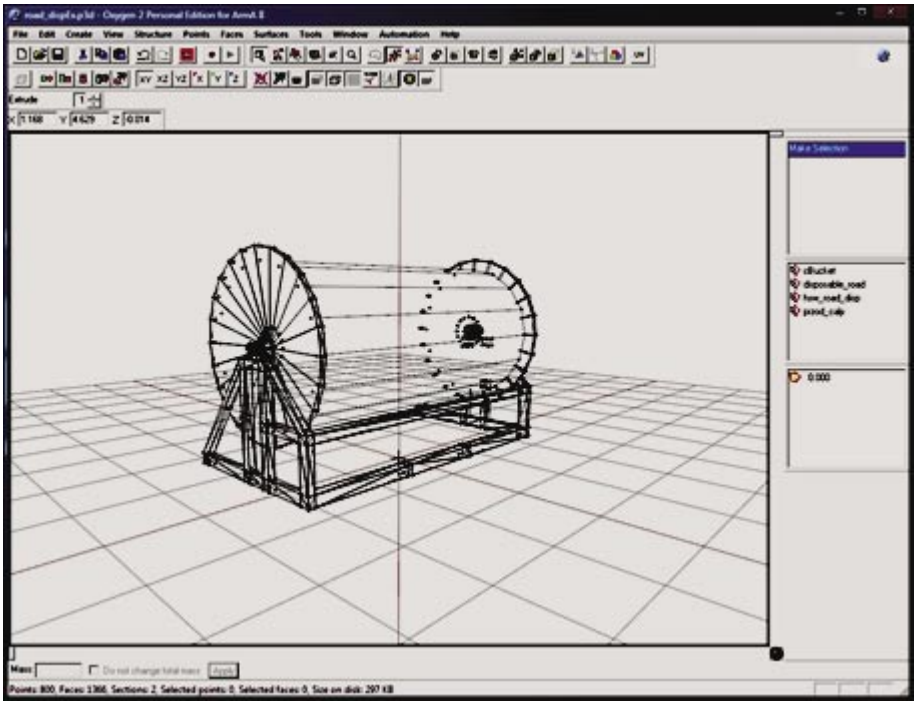
2.1. Kompilacja

Hydrogen nie jest kompilatorem. Kompilacja skryptów VBS, tj. programów napisanych w języku SQF, odbywa się w narzędziu VBS o nazwie Addon Packer [7]. Hydrogen stara się w jak największym stopniu ułatwić pracę ze skryptami VBS. Co więcej, po odpowiednim przygotowaniu ścieżek programu Addon Packer możliwe jest uzyskanie ewentualnych informacji o błędach w kodzie źródłowym po wciśnięciu przycisku *Start packing!* (rys. 2), kiedy to następuje pakowanie zasobów i wykrywanie błędów - informacje podawane są w konsoli programu.



Rys. 2. Ekran główny aplikacji Addon Packer systemu VBS

Niestety w większości przypadków pracy nad nowymi funkcjonalnościami systemu VBS korzystamy z dodatkowych zasobów, np. modeli 3D utworzonych w programie Oxygen [8] (rys. 3), i w zależności od stopnia skomplikowania modelu, kompilacja i pakowanie zasobów może trwać nawet kilka minut.



Rys. 3. Ekran główny aplikacji Oxygen z przykładową edycją obiektu 3D

2.2. Interpretacja, leksykacja i tokenizacja

Autor podjął próbę interpretacji kodu języka SQF i plików konfiguracyjnych w celu wykrywania błędów już z poziomu środowiska Hydrogen, jednak stopień zawiłości i brak

dostępu do jądra systemu VBS, skutecznie to uniemożliwił. System skryptów VBS jest w ciągłej fazie rozwoju, ma unikalną składnię (to nie jest zaleta), a sam język SQF nie jest językiem obiektowym – posiada cechy języka strukturalnego. Wszystkie te cechy ograniczyły funkcjonalność środowiska Hydrogen do analizy leksykalnej[3] i tokenizacji skryptów SQF i plików konfiguracyjnych. Dzięki udostępnieniu przez firmę Bohemia Interactive informacji na temat kategorii (tokenów) języka SQF i skryptów konfiguracyjnych [9], przygotowano odpowiednie struktury (tablica 1, 2, 3), które stały się głównym źródłem informacji dla systemu kolorowania składni w środowisku Hydrogen.

Tablica 1. Tokeny plików konfiguracyjnych i skryptów SQF

Token – reprezentowana kategoria	Opis lub przykładowe leksemy
Słowo kluczowe (keyvalue)	class
Identyfikator (identifier)	Nazwy zmiennych wprowadzone przez programistę
Liczba lub tekst (number or string)	Stałe liczbowe i tekstowe, np. 1, 20, 'tekst'
Komentarz (comment)	Wyrażenia po // i pomiędzy /* a */
Komenda preprocesora (preprocesor command)	#define, #ifdef
Pole edycyjne[1]	Pola do uzupełnienia oznaczone trzema kropkami

Tablica 2. Dodatkowe tokeny języka SQF

Token – reprezentowana kategoria	Przykładowe leksemy
Komenda sterująca (control command)	and, for, or, not
Komenda regularna (regular command)	setMarkerColor, setMass, sleep
Funkcja (function)	fn_vbs_absSpeed, fn_vbs_vectorAdd

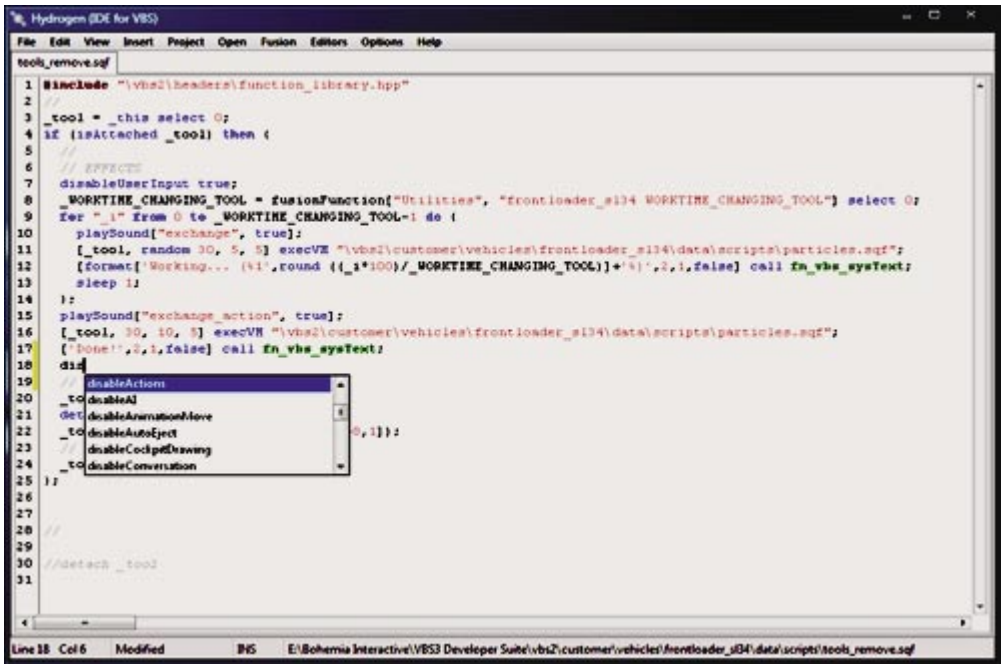
Tablica 3. Dodatkowe tokeny plików konfiguracyjnych

Token – reprezentowana kategoria	Przykładowe leksemy
Właściwość konfiguracyjna (config property)	additionalSound, advancedWeaponSelection
Prefiks właściwości konfiguracyjnej (config property prefix)	actionBegin, soundHit
Zdarzenie dialogowe (dialog event)	onChanged, onKeyDown
Właściwość zdarzenia dialogowego (config dialog event)	colorText, animPeriod
Nazwa klasy (class)	Turrets, Grid, Sounds

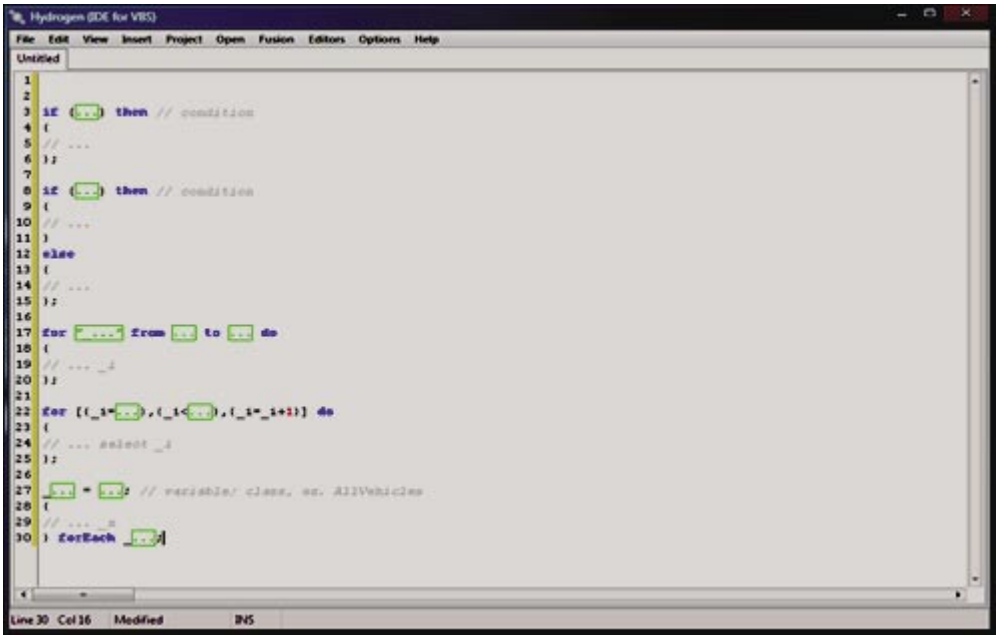
[3]Analiza leksykalna kodu źródłowego to grupowanie pojedynczych znaków w odpowiednie tokeny (kategorie) [4].

2.3. Edytor kodu źródłowego

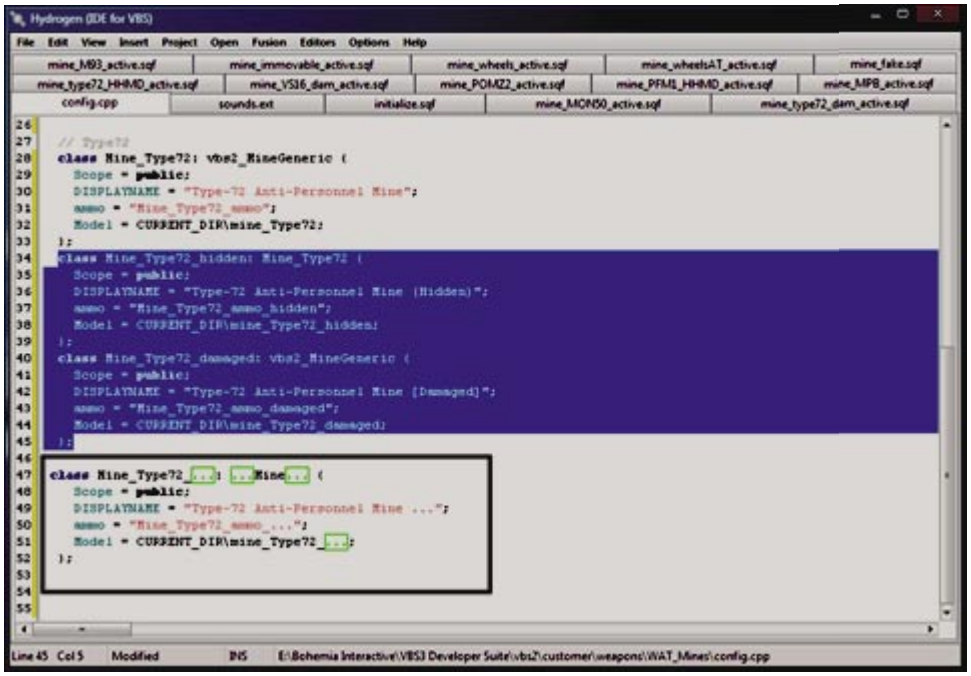
Kolorowanie składni to niejedyna zaleta edytora kodu źródłowego środowiska Hydrogen. Dużą uwagę poświęcono systemowi podpowiedzi, który posiada pełną listę możliwych do użycia metod (rys. 4) i służy do automatycznego uzupełniania tekstu (rys. 5 i 6).



Rys. 4. Ekran główny edytora kodu źródłowego w środowisku Hydrogen z widoczną listą podpowiedzi (skrót klawiszowy: [ctrl+space])

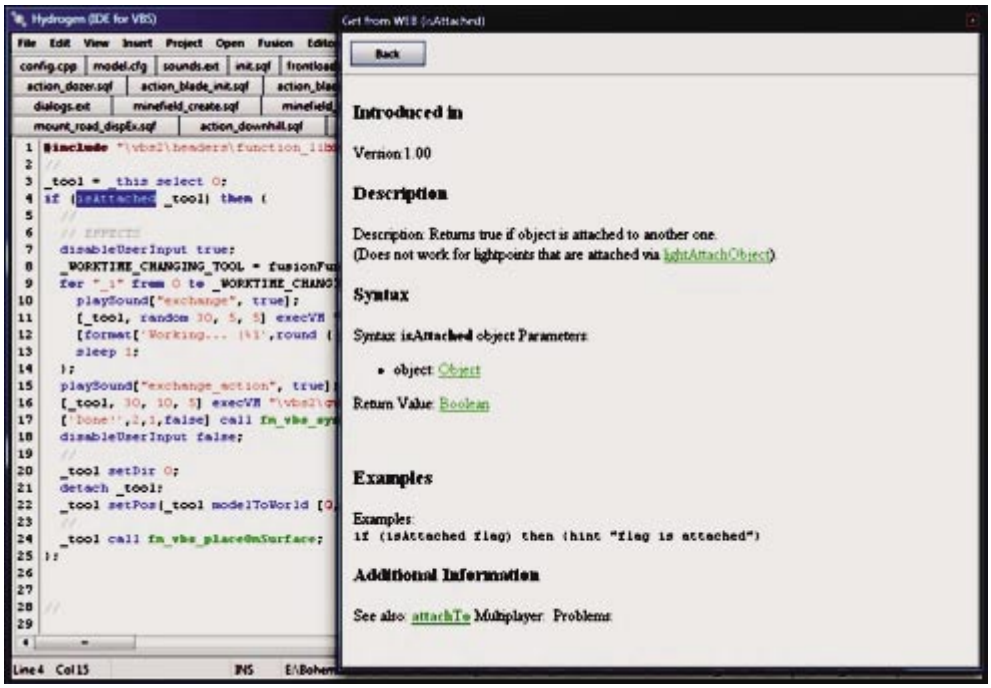


Rys. 5. Ekran główny edytora kodu źródłowego z automatycznie uzupełnioną składnią SQF ([shift+space] dla przykładowych wyrażeń: if, ifelse, for, fori, foreach)



Rys. 6. Poglądowy widok kopiowania struktury klasy z wykorzystaniem algorytmu obliczania odległości Damerau-Levenshteina [10] (skrót [ctrl+D] kopiuje strukturę klasy na podstawie jednej lub więcej klas)

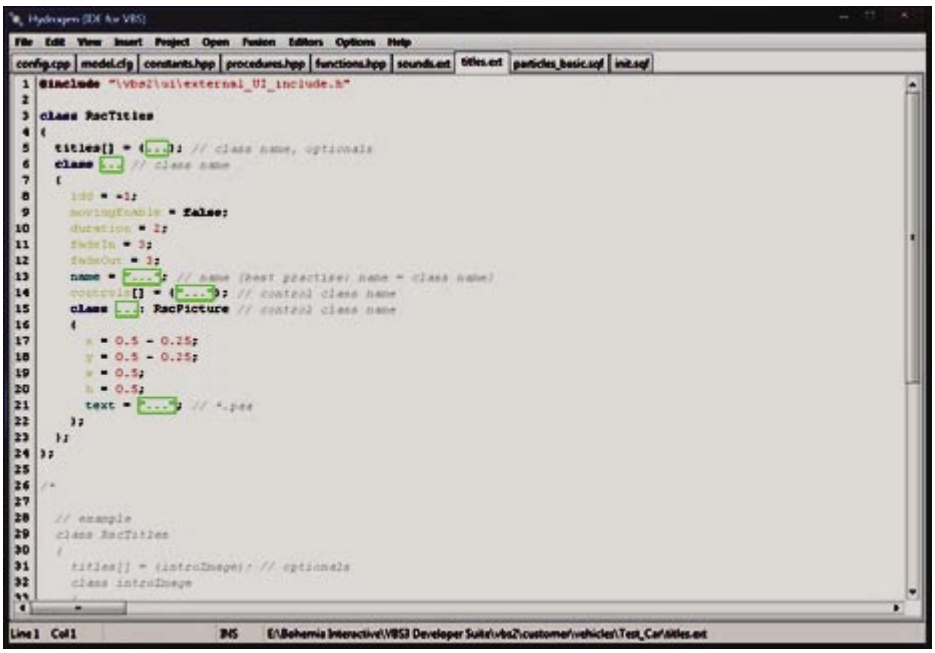
Opis składni prezentowany w postaci przeglądarki HTML (rys. 7) można uzyskać poprzez zaznaczenie dowolnego wyrażenia i wciśnięcie skrótu [F1]. Informacje pobierane są ze strony *https://resources.bisimulations.com* [11] z wykorzystaniem protokołu SSL (biblioteka OpenSSL) [12].



Rys. 7. Ekran główny przeglądarki HTML (skrót [F1] po zaznaczeniu wyrażenia w kodzie źródłowym)

2.4. Szablony obiektów i wtyczki fusion

Środowisko programistyczne Hydrogen powstawało w Biurze Symulatorów OBRUM sp. z o.o. wraz z opracowywaniem dla Wojskowej Akademii Technicznej projektu, który obejmował wykonanie i dodanie do systemu VBS kilkunastu pojazdów, kilkudziesięciu min i ładunków IED wraz z ręcznymi wykrywaczami min. W celu przyspieszenia prac nad projektem, środowisko Hydrogen rozbudowano o szablony dla tych rodzajów obiektów. Rysunek 8 przedstawia przykładowy projekt pojazdu, utworzony przez szablon pojazdu, na który składa się dziewięć podstawowych plików. Wykorzystanie szablonów znacznie przyspieszyło i uprościło pracę oraz narzuciło pewien estetyczny podział kodu źródłowego, który może być zrozumiały dla mniej doświadczonych programistów.

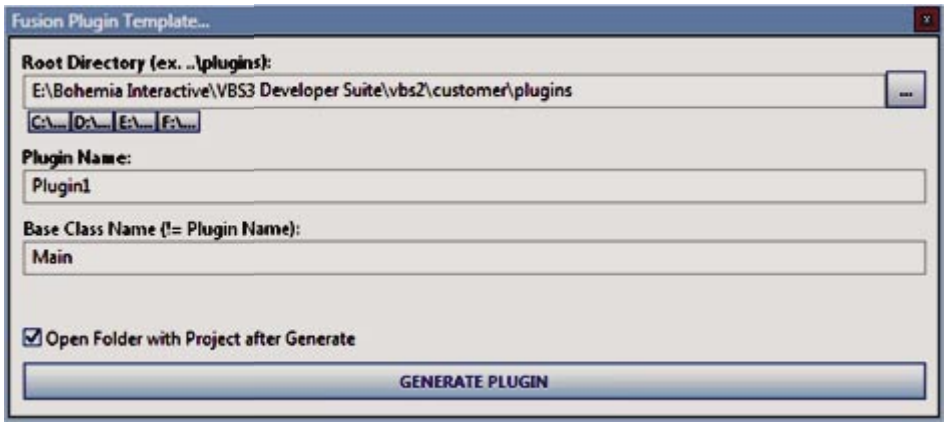


Rys. 8. Ekran główny edytora kodu źródłowego z przykładowym plikiem konfiguracyjnym, utworzonym przez szablon pojazdu środowiska Hydrogen (wyraźne pola edycyjne wskazują miejsca do uzupełnienia przez programistę)

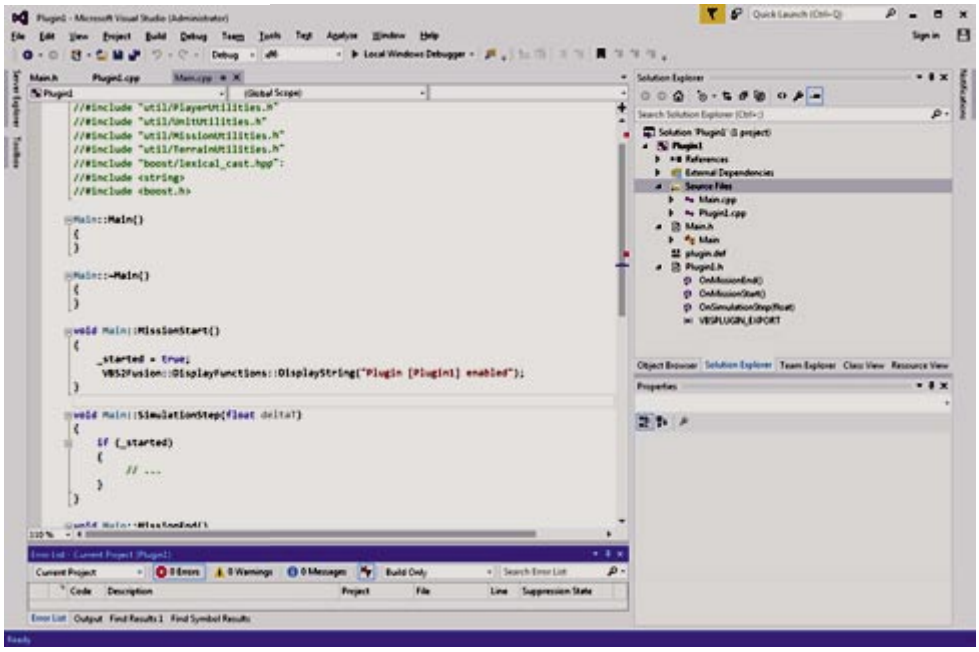
Nie każdą funkcjonalność da się wprowadzić do systemu VBS za pomocą skryptów i plików konfiguracyjnych. Operacje na obiektach, zdefiniowanych w systemie VBS, można wykonywać również za pomocą wtyczek Fusion[4] [13]. Wtyczka Fusion, czyli biblioteka DLL, jest tworzona w oparciu o zestaw klas C++ i stanowi zewnętrzne rozszerzenie systemu [14]. Kompilację wtyczek Fusion można przeprowadzić w środowisku programistycznym Microsoft Visual Studio [15].

Hydrogen umożliwia przygotowanie projektu wtyczki dla środowiska Visual Studio (rys. 9), który może być natychmiast skompilowany do biblioteki DLL (rys. 10).

[4]Skrypty SQF i wtyczki Fusion bardzo dobrze uzupełniają swoje braki. Nie jest jednak możliwe korzystanie tylko z jednej lub drugiej metody w bardziej wymagających projektach.



Rys. 9. Okno edycyjne służące do generowania projektu wtyczki Fusion w środowisku Hydrogen

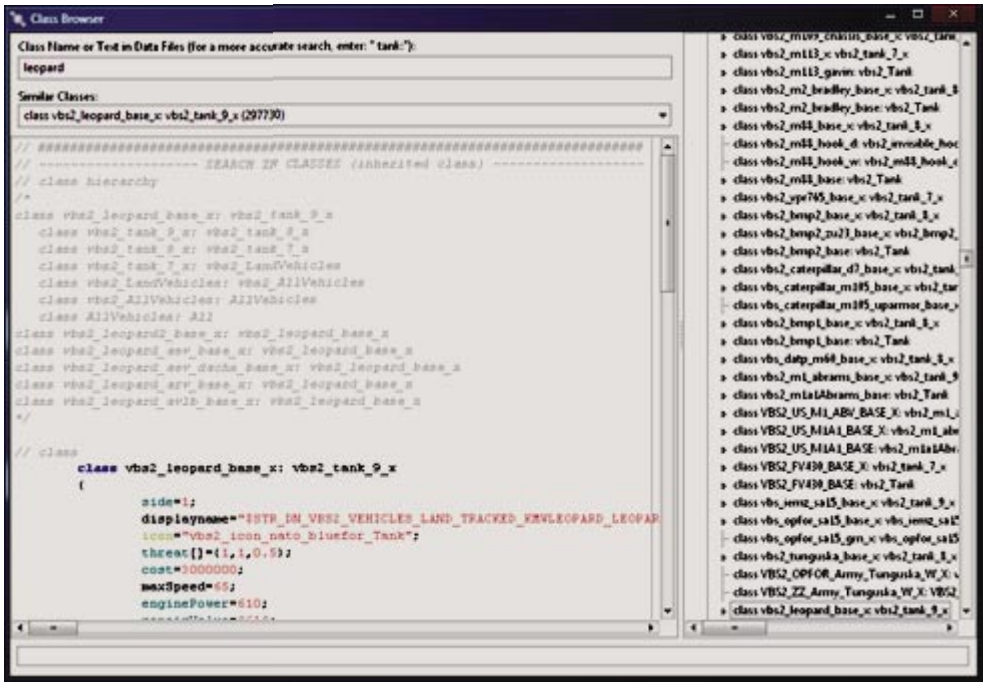


Rys. 10. Ekran główny środowiska Visual Studio z kodem źródłowym wtyczki Fusion przygotowanym w środowisku Hydrogen

2.5. Przeglądarka klas

System VBS zawiera ogromną bazę obiektów (klas) opisanych za pomocą skryptów SQF i plików konfiguracyjnych. Informacje o klasach (właściwości klas oraz ich parametry), znajdujące się w pliku *AllConfig.cpp* i *CfgVehicles.txt* [16], umożliwiły opracowanie systemu przeglądania klas z wyszukiwarką (rys. 11). Narzędzie zawiera ponadto drzewo dziedziczenia klas, które ułatwia znalezienie wszystkich parametrów danej klasy[5].

[5]Dziedziczenie to mechanizm współdzielenia funkcjonalności pomiędzy klasami, np. klasa *soldier* (podklasa) dziedziczy funkcjonalności po klasie *człowiek* (klasa bazowa). Aby znaleźć wszystkie właściwości i parametry danej klasy w systemie VBS, należy przeanalizować klasy bazowe danej podklasy.



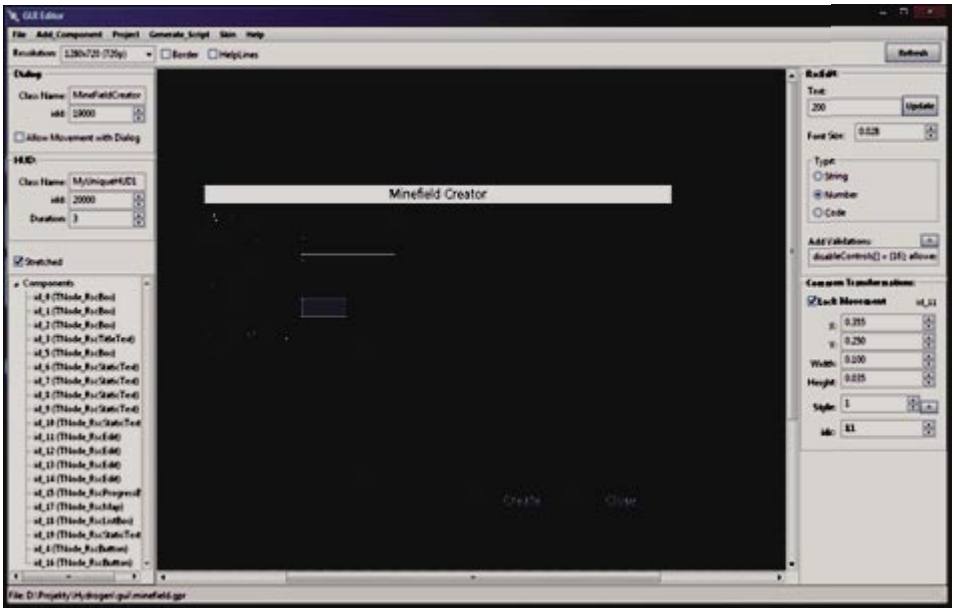
Rys. 11. Ekran główny przeglądarki i wyszukiwarki klas VBS w środowisku Hydrogen

2.6. Edytor interfejsu użytkownika

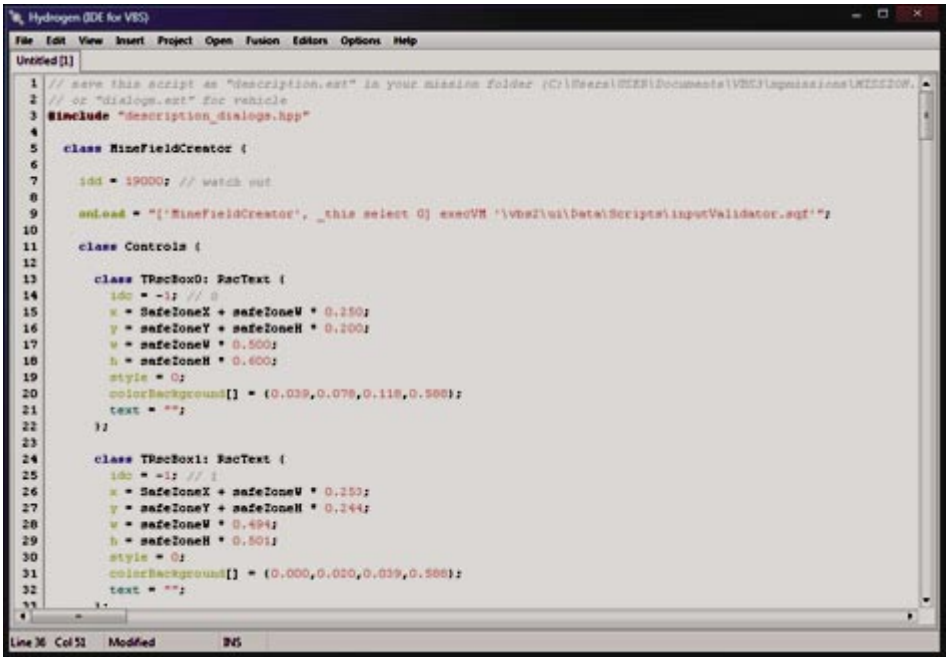
Projektowanie graficznego interfejsu użytkownika za pomocą plików konfiguracyjnych (pliki tekstowe) systemu VBS to bardzo czasochłonny, skomplikowany i nienaturalny[6] proces. Autor nie znalazł informacji na temat istniejących edytorów interfejsu użytkownika dedykowanych systemowi VBS i sam system takiego nie posiada (lub nie jest dostarczany).

Opracowanie i zaimplementowanie wizualnego edytora GUI (ang. *Graphic User Interface*) do przygotowywania graficznego interfejsu użytkownika systemu VBS (rys. 12) stało się ważnym czynnikiem przyspieszającym i ułatwiającym pracę. Edytor pozwala na swobodne dodawanie i manipulację komponentami wizualnymi[7], takimi jak: RscButton, RscBox, RscText, RscEdit, RscPicture, RscTitleBar, RscTrackBar, RscProgressBar, RscToolBox, RscListBox, RscComboBox, RscMap oraz automatyczne generowanie plików konfiguracyjnych (rys. 13).

[6]Edytory typu WYSIWYG (ang. *WhatYouSeeIsWhatYou Get*) to w obecnych czasach standardowe systemy edycji kontentu - pozwalają na podgląd efektu końcowego w czasie projektowania.
[7]Komponenty wizualne opatrzono nazwami według konwencji systemu VBS.



Rys. 12. Ekran główny wizualnego edytora interfejsu użytkownika w środowisku Hydrogen z przykładowym interfejsem generującym pole minowe



Rys. 13. Ekran główny edytora tekstu z automatycznie wygenerowanym plikiem konfiguracyjnym

Efekt końcowy, tzn. interfejs widoczny w symulacji VBS (rys. 14), prawie całkowicie pokrywa się z interfejsem zaprojektowanym w środowisku Hydrogen – różnice mogą dotyczyć czcionki i braku dynamicznych elementów graficznych w edytorze (w prezentowanym przykładzie jest to mapa GPS).

Dodatkowe funkcjonalności edytora to możliwość zaznaczenia wielu komponentów wizualnych (szeregowanie, wyrównywanie), możliwość podglądu interfejsu w różnych rozdzielczościach, rysowanie linii pomocniczych, zmiana schematów kolorów.

Do implementacji komponentów wizualnych edytora interfejsu w środowisku Hydrogen wykorzystano bibliotekę graficzną Graphics32 [17].

Rys. 14. Interfejs użytkownika w uruchomionej symulacji VBS

3. PODSUMOWANIE

W artykule opisano podstawowe cechy i właściwości środowiska programistycznego Hydrogen. Artykuł dedykowany jest programistom, którzy nie mają dużego doświadczenia z systemem VBS i pozwala w przystępny sposób zapoznać się z nowym narzędziem. Pokazuje trudności, z jakimi należałoby się zmierzyć pracując z systemem VBS przy braku podobnego środowiska (np. projektowanie graficznego interfejsu użytkownika za pomocą skryptu konfiguracyjnego, który jest plikiem tekstowym) i w sposób pośredni sygnalizuje zwrot nakładu pracy i czasu, jaki można uzyskać przez wykorzystanie chociażby szablonów.

Środowisko programistyczne Hydrogen nie wyróżnia się wśród nowoczesnych środowisk programistycznych, ale jest przykładem wykorzystania wiedzy i technologii programistycznych w zakresie metod ich powstawania w celu wygodniejszego tworzenia oprogramowania dla systemu symulacji Virtual Battlespace. Z punktu widzenia oferowanych na rynku narzędzi dedykowanych systemowi symulacji VBS, Hydrogen jest narzędziem nowym i nie zawiera żadnych zamkniętych czy komercyjnie licencjonowanych źródeł. Stopień użyteczności praktycznej środowiska Hydrogen eliminuje konieczność używania ograniczonego środowiska Script Debugger Plugin dostarczanego z deweloperską wersją VBS, a samo środowisko zapewnia tworzenie skryptów w sposób szybki, uporządkowany i estetyczny. Opracowanie środowiska Hydrogen w Biurze Symulatorów OBRUM sp. z o.o. zapewnia ciągłość rozwoju oprogramowania (pełny kod źródłowy) przy regularnej ewolucji systemu VBS.

Najważniejsze elementy środowiska Hydrogen to zaawansowany edytor kodu źródłowego i plików konfiguracyjnych, przeglądarka klas systemu VBS oraz edytor interfejsu graficznego, który umożliwia projektowanie interfejsu z wykorzystaniem komponentów wizualnych.

4. DODATEK. Narzędzia środowiska Hydrogen - główne cechy

Cechy edytora tekstowego:
- edytor kodu źródłowego z kolorowaniem składni języka skryptowego SQF (format *.sqf);
- edytor pików konfiguracyjnych z kolorowaniem składni plików w formacie *.cfg, *.ext, *.hpp, *.config, *.cpp;
- wczytywanie i zapisywanie projektów z edycją listy plików wchodzących w skład projektu;
- możliwość tworzenia kopii zapasowych projektów (pakowanie plików do formatu *.zip);
- uzupełnianie słów i składni SQF;
- lista słów kluczowych języka SQF i plików konfiguracyjnych;
- kopiowanie struktury klas;
- szablony obiektów;
- tworzenie szablonów dla wtyczek Fusion;
- systemy pomocy (wbudowana przeglądarka HTML).

Cechy systemu przeglądania klas:
- wyszukiwarka klas;
- drzewo hierarchii klas;
- kolorowanie składni w przeglądarce i dołączony system pomocy.

Cechy edytora interfejsu graficznego:
- edytor typu WYSIWYG;
- kilkanaście komponentów wizualnych;
- kilkanaście schematów graficznych;
- swobodna edycja komponentów wizualnych;
- generowanie skryptów VBS na podstawie zaprojektowanego interfejsu;
- skalowalność – możliwość podglądu interfejsu w różnych rozdzielczościach ekranu z zachowaniem proporcji lub bez (parametr *stretch* w systemie VBS).

5. LITERATURA

[1] Bohemia Interactive: VBS3, https://bisimulations.com/virtual-battlespace-3 [dostęp: 18.09.2016].
[2] Bohemia Interactive: VBS3 NATO, https://nato.bisimulations.com/ [dostęp: 18.09.2016].
[3] Bohemia Interactive: Script Debugger Plugin, https://manuals.bisimulations.com/vbs3/3-6/manuals/Content/Editor_Manual/SM_ScriptDebugger.htm [dostęp: 19.09.2016].
[4] Aho A.V., Sethi R. Ullman J.D.: Compilers: Principles, Techniques, and Tools, 1986, Addison Wesley, ISBN-10 0-321-48681-1.
[5] Free Pascal Team: Free Pascal, http://www.freepascal.org/ [dostęp: 20.09.2016].
[6] Lazarus and Free Pascal Team: Lazarus, http://www.lazarus-ide.org/ [dostęp: 20.09.2016].
[7] Bohemia Interactive: Addon Packer, https://manuals.bisimulations.com/vbs2/2-00/devref/Content/Adding_Models/AM_Packing.htm [dostęp: 21.09.2016].

[8] Bohemia Interactive: Oxygen Manual, https://manuals.bisimulations.com/vbs3/3-4/devref/Content/Oxygen_Manual/Oxy_Oxygen_Manual.htm [dostęp: 21.09.2016].
[9] Bohemia Interactive: Text Editors, https://resources.bisimulations.com/wiki/Text_editors [dostęp: 22.09.2016].
[10] Damerau F. J.: A technique for computer detection and correction of spelling errors, 1964, http://portal.acm.org/citation.cfm?id=363994 [dostęp: 20.05.2016].
[11] Bohemia Interactive: VBS Scripting Reference, https://resources.bisimulations.com/wiki/Main_Page [dostęp: 21.05.2016].
[12] OpenSSL Software Foundation: Welcome to OpenSSL! ,https://www.openssl.org/ [dostęp: 21.05.2016].
[13] SimCentric Technologies, VBSFusion, https://www.simct.com/products/vbs3/vbsfusion [dostęp: 16.08.2016].
[14] Wantoch-Rekowski R.: VBS2: Programowalne środowisko symulacji wirtualnej, 2015, PWN, ISBN 978-83-01-18170-3.
[15] Microsoft, Visual Studio 2013, https://msdn.microsoft.com/pl-pl/library/dd831853(v=vs.120).aspx [dostęp: 17.08.2016].
[16] Bohemia Interactive: Startup Parameters, https://community.bistudio.com/wiki/Virtual_Battlespace:_Startup_Parameters [dostęp: 17.08.2016].
[17] Graphics32 Team: Graphics32, http://graphics32.org/wiki//Main/HomePage [dostęp: 4.09.2016].

HYDROGEN – A DEVELOPMENT ENVIRONMENT FOR THE VIRTUAL BATTLESPACE SYSTEM

Abstract. The article presents the capabilities and technological solutions of the Hydrogen programming environment devised at OBRUM's Simulator Department and intended for the Virtual Battlefield 3 system. Hydrogen is in addition to the tools provided by Bohemia Interactive, the VBS system manufacturer, and facilitates expanding the system with new functionalities.

Keywords: Virtual Battlespace, development environment, programming, scripting language, GUI editor.