THE RISK MANAGEMENT ASSOCIATION
Serving the Financial Services Industry

1801 Market Street, Suite 300, PHILADELPHIA, PA 19103-7398 • 215-446-4000 • FAX 215-446-4101 • www.rmahq.org

August 2, 2010

Elizabeth M. Murphy,
Secretary, Securities and Exchange Commission,
100 F Street, NE, Washington, DC 20549-1090

Delivered via e-mail to rule-comments@sec.gov

RE: File Number S7-08-10, *Response to Waterfall Computer Program (Section III.B.1)*

Ladies and Gentlemen:

Thank you for providing The RMA Securitization Risk Roundtable (SRR) the opportunity to respond to the SEC's request for comment on the 'Filing of a Computer Program That Gives Effect to the Waterfall Provisions of the Transaction' (Section III.B.1), published on 7 April 2010 as one of the proposed rules to amend the offering process and disclosure and reporting requirements for ABS. The SRR consists of a group of risk professionals who meet on a regular basis to discuss key aspects of risk modeling and securitization, with a focus on improved risk evaluation and transparency. Additional information about SRR is available at http://www.rmahq.org/RMA/MarketRisk/SRR.htm and SRR can be contacted at securitization@rmahq.org.

Introduction and Overview

SRR strongly supports the Waterfall Computer Program (WCP) proposal whereby most ABS issuers would be required to file a computer program that gives effect to the flow of funds, or "waterfall", provisions of the transaction. In particular, we agree that the WCP will help accomplish the following two key objectives as described in the proposal:

(i) 'Make it easier for an investor to conduct a thorough investment analysis of the ABS offering at the time of its initial investment decision.' (page 206, III.B.1)
(ii) 'Monitor ongoing performance of purchased ABS by updating its investment analysis from time to time to reflect updated asset performance.' (page 206, III.B.1)

In the lead-up to the recent global financial crisis, market participants were faced with high transaction volumes and an ever-increasing range of complicated waterfall structures. It has become clear that many of these market participants relied very heavily on the opinions of third party vendors to make investment decisions. The heavy reliance was driven by the lack of readily available, affordable tools that would have helped to facilitate an independent and more thorough analysis performed by the investors directly. Given the complex and historically closed nature of ABS structures, we believe that there is a

pressing demand for a new way to look at these securities and to provide solutions that would facilitate clear, detailed and relevant pricing and risk analysis made readily available to all investors, both large and small.

The requirement to file the WCP would be a great stride in the right direction. The idea that 'market participants would be able to conduct their own evaluations of ABS and may be less dependent on the analysis of third parties such as credit rating agencies' is an important direct benefit. All investors, large and small, would benefit from this new, more open approach. Tools and data required to perform appropriate analysis would be attainable even for smaller institutional investors or others who are under cost constraints. In addition, investors would not have to rely on issuers' computational materials to form their investment view, but rather would have the opportunity to formulate an independent view. We think that the new proposed approach will result in significant improvements in both transparency of the ABS issuing process and availability of appropriate tools (to ALL investors) to undertake detailed and relevant valuation and risk analysis.

Investors would spend less time chasing the English language statement of the waterfall within the deal documents and going through the steps to convert to computer code in a programming language and more time analyzing and understanding the transactions. Moreover such analysis and understanding could be achieved on a timely basis and with less likelihood for errors. We believe that this would create a more level playing field for all investors.

Regarding the choice of language, we believe that Python is a reasonable choice of language for implementing the WCP. With Python as the choice, we point out that in order for the WCP to be most useful and fulfill its full potential, it will be important to ensure that the industry agrees on common software architecture across all WCP submissions. For example, there should be an established and agreed upon standard set of input and output interfaces by product and asset class. We would suggest the formation of industry groups to establish the most appropriate/useful interfaces for each asset class. One suggestion on how this could be successfully accomplished would be via an open-source class library, which could be controlled by a not-for-profit organization, which represents all stakeholders, large and small. The organization would be given the responsibility to set the standards, implementation, documentation, etc. of the class library in a manner that is consistent with the needs of investors, industry players and regulators.

Points of emphasis

With these overarching remarks, the RMA (SRR) would like to provide additional comments on several of the questions where we believe the recommendations could be improved to enhance their effectiveness.

Comments to specific questions include:

1. Is it appropriate for us to require most ABS issuers to file the waterfall computer program? Is there an alternative form of required information filing that would be more useful to investors, subject to the limitation that executable code may not be filed on EDGAR?

SRR supports requiring ABS issuers to file a waterfall computer program (WCP). It allows investors to gain a better understanding of the ABS structure and makes it easier for investors to run *what if* scenarios as part of their due diligence process.

While the requirement that a WCP in a computer language such as Python would be a definite improvement over the existing process and approach, requiring that the waterfall structure be described in a simple declarative manner would be even more useful to investors because it would make understanding the structure easier and would make it possible for third parties to build pricing and risk management tools.

2. Should we require, as proposed, that the Rule 424(h) filing include the waterfall computer program?

Yes.

3. Does access to the waterfall computer program decrease the amount of time needed to analyze the information in a prospectus?

Definitely, the WCP requirement will allow investors to analyze new structures more efficiently. They can skip the reverse engineering of deals or avoid waiting for third party vendors to model the deal and make it available to investors.

4. If we adopt the waterfall computer program filing requirement, would less time be needed for investors to review transaction-specific information?

Investors will spend more time analyzing transactions and less time chasing information from originators or trustees.

5. If so, how much time would be needed after the waterfall computer program is filed? Four days? Two days? Does analysis of the waterfall computer program require more time than what we allow as proposed so that we should increase the time period for the Rule 424(h) filing?

This depends on the complexity of the ABS structure and the level of sophistication of the investor.

6. Is it appropriate to require issuers to submit the waterfall computer program in a single programming language, such as Python, to give investors the benefit of a standardized process? If so, is Python the best choice or are there other open-source programming language alternatives (such as Perl) that would be better suited for these purposes?

If the goal of the waterfall computer program (WCP) is to enhance the transparency of the ABS issuing process, then it would indeed be a good idea to require that all WCPs be written in the same language. In this way, investors who wished to examine the source code would only have to be familiar with one language, and anyone who wanted to run the code, including third-party application developers, would only have to set up one runtime environment for this purpose.

Python is a superb choice of language for WCPs because it is concise and expressive. Like all dynamic scripting languages, including Perl and Ruby, it allows developers to write programs that will run without further modification on any computing platform where the language interpreter has been installed. Python's syntax contains few of the cryptic constructs that abound in Perl, and it is less reliant than Ruby on exotic functional-programming concepts. Investors are therefore likely to find Python more readable than any competing language.

7. Should more than one programming language be allowed? If so, which ones and why?

It would be ideal if a single specified language and no other were allowed for the submission of WCPs. Permitting different languages would make it more difficult to compare WCPs with one another, both in terms of human readability and software compatibility. ABS issuers who internally develop their software in a different language should be responsible for expressing each WCP in the lingua franca rather than transferring this burden to investors or third parties. Because the WCP is intended to be the authoritative public description of an ABS issue, its submission format should be standardized as far as possible in order to maximize its accessibility to investors.

The WCP should be singular and comprehensive, meaning that users should be able to run it without relying on any other software apart from the program execution environment (i.e., interpreter or compiler), and it should fully compute the flow of funds without relying on any other information apart from the input data. The use of a programming language ensures that the waterfall can be captured in its entirety. Programming languages are Turing-complete, so according to the Church-Turing thesis, any computable function (hence any flow of funds) can be computed with a programming language. This is one great advantage of describing waterfalls with a programming language rather than with a data description language.

The other great advantage is that a program is unambiguous. Unlike static data descriptions which require further explication to determine the precise flow of funds, the program describing a waterfall can be directly executed to determine the flow of funds for a given input. The program itself, together with the specification of the language in which it is written, suffices as a complete and unambiguous description of the flow of funds. Resorting to any additional representation of the waterfall would only introduce ambiguity.

8. Should we restrict ourselves to only open source programming languages or allow fully commercial or partly-commercial languages (such as C-Sharp or Java) to be used? If so, what factors should be considered?

We believe that the WCP should be written in a mature language that is fully defined in a publicly accessible specification document. This is not quite the same thing as *open source*, a term which, strictly speaking, applies to software and not to abstract specifications. On the other hand, it would be advantageous to use a programming language that has been implemented in open-source software, which means that it is freely available to the public and that it is open to review by anyone with sufficient expertise. We must note that there is a wide variety of open-source licenses, some of which are less friendly to commercial usage than others. The most widely used Python interpreter is distributed under a license that allows unlimited use without charge for any purpose, including commercial purposes.

In choosing a programming language for an open marketplace, the most important factors to consider are transparency, accessibility, and standardization. In the case of Python, the requirements of transparency and accessibility are fulfilled by the Python Language Reference, which can be viewed in its entirety online, and by the Python implementations that can be downloaded and used by anyone at zero cost. As for standardization, it will be important for the SEC to specify not just what language WCPs are to be written in, but which version of the language. Like every popular programming language, Python is under continuous development, with a new language specification

published periodically under a new version number. In order for all WCPs to be fully compatible with one another, they must all be written in the same version of Python.

We recommend that the SEC stipulate the use of Python 3, which departs in significant ways from Python 2. More specifically, we recommend that the SEC require that every WCP comply with the Python 3.1.2 language definition. The simplest way to achieve such compliance is to use a Python 3.1.2 interpreter in the development process. In practice the differences among minor version numbers are so small that one can expect a program developed under version 3.1.2 to run correctly under version 3.2.0, for example, and vice versa. Nonetheless, we feel that the version number should be precisely specified in order to minimize the possibility of confusion. We also recommend that the required version number be frozen for a significant period of time, such as five or ten years, so that all WCPs published during that time will be compatible with one another.

9. Are there other requirements we should impose on the possible computer programming languages that are used to satisfy this requirement, other than that such languages be open source and interpreted?

We have discussed above the general merits of open source as well as the necessity of imposing a version requirement on the programming language to be used in WCPs. Let us now turn our attention to the question of interpretation.

First of all, it is incorrect to classify a programming language as either interpreted or compiled. Such a characterization can only be made of an implementation rather than of a language itself. Interpretation means that a program is executed one statement at a time, while compilation means that the whole program is translated into another language—generally assembler or virtual-machine code, but potentially any language—prior to execution. Any programming language can be either interpreted or compiled, which implies that no programming language is inherently an interpreted or a compiled language. For example, even though the most widely used implementation of Python, known as CPython, is an interpreter, there is a popular alternative implementation called Jython that compiles Python into Java Virtual Machine instructions.

The use of an interpreter rather than a compiler to run a program does not make it any safer. We agree that it would be unwise to allow binaries (i.e., machine code) to be disseminated through EDGAR, because it is difficult or impossible to determine what a binary is designed to do. But as long as a program is published in the form of source code, it makes no difference with respect to security whether the end user chooses to interpret or compile the program. The chief benefit of using an interpreter is ease of execution. In the case of Python, the ready availability of a standard Python interpreter means that users can easily run a Python program in one step, without undertaking the compilation process that is typically used for programs written in languages like C++.

So the important distinction to make is between binary and source code rather than between interpretation and compilation. We fully agree that the WCP should be published as source code for the sake of transparency and security. On the other hand, we are concerned that parts of the SEC proposal draw a false dichotomy between interpretation and compilation. The text on page 23380 of the Federal Register (Vol. 75, No. 84) equates the term *executable code* with binary code and suggests that interpretation provides a safe harbor from security risks. In fact, this text contradicts itself with footnote 353, which points to the definition in Rule 11 of Regulation S-T: "The term

executable code means instructions to a computer to carry out operations that use features beyond the viewer's, reader's, or Internet browser's native ability to interpret and display HTML, PDF, and static graphic files. Such code may be in binary (machine language) or in script form." Script form is source code, which means that the SEC's own regulation prohibits the submission of any program that can run outside the browser, whether through an interpreter or otherwise.

The SEC's own regulation prohibits the dissemination through EDGAR of any program written in a general-purpose programming language. This means that the WCP proposal cannot be implemented under the regulations as they currently stand. We therefore urge the SEC to strike Rule 106 and replace it with one that prohibits the submission of binary code, while permitting the submission of source code in approved languages.

10. Under our proposal, issuers would be required to file the waterfall computer program in the form of downloadable source code on EDGAR. Prior to filing, the code would not be tested by the Commission. Would downloading the code onto a local computer give rise to any significant risks for investors? If so, please identify those risks and what steps or measures we should take to address the risks, if any.

There is always a risk when users download and run arbitrary code. This is especially true for dynamic scripting languages such as Python, which are designed to give programmers broad access to underlying system resources. One can easily write a script that reads and manipulates the user's files, uploads or downloads data through the Internet, and executes other programs on the user's account.

In theory, one can safeguard against malicious code by using a restricted execution environment, known as a sandbox, to run programs. A sandbox isolates the program it is running from the general resources of the computer system, so that potentially harmful instructions execute without effect. But even if the SEC were to provide a sandbox for WCPs—and we are not suggesting that it should—nothing would prevent users from running the code in an unrestricted interpreter outside the confines of the sandbox and putting their systems at risk.

The best protection for investors is the fact that not just anyone can publish source code on EDGAR. The EDGAR system restricts access to registered entities whose actions on the system are logged in detail. Users should be warned that the code they download from EDGAR has not been tested by the SEC and has the potential to manipulate all resources on a computer system where it is executed, including any valuable data that may be stored on it. On the other hand, ABS issuers should be advised that they are liable for any malicious code in their WCPs, just as they are liable for inaccuracies in their prospectuses.

If the onus is on issuers to provide innocuous code, they will be motivated to develop clear and concise programs that they can easily vet for code safety, thereby furthering the transparency of the WCP. In order to reassure themselves that their code is unlikely to harm a user's computer system, issuers could be encouraged by the SEC, without compulsion or enforcement, to follow a set of voluntary coding guidelines that serve to isolate a program from general system resources.

In the case of Python, these coding guidelines would include the following restrictions:

- Avoid the use of the sys and os modules, which have facilities for manipulating files and processes.

- Do not use the open() instruction to read or write any file, and do not use any other facility to make file objects.
- Avoid the use of urllib, ftplib, and all other modules that make network connections.

WCP developers who follow such a set of guidelines can be confident that their code is very unlikely to damage a computer system upon which it is executed. In addition to checking the code manually, issuers could potentially use an automatic verification tool to determine whether a WCP satisfies the safe coding guidelines. But again, any such efforts should be voluntary and not mandated by the SEC. We support the principle that the issuer should be liable for any malicious effects of a WCP and that it should be the issuer's responsibility, not the SEC's, to ensure that their WCPs do not contain malicious code.

11. Are the proposed input and output requirements for the waterfall computer program appropriate? If not, what type of output and tests should be required for the waterfall computer program? Should the outputs of the waterfall computer program be specified in detail by rule, or broadly defined to afford flexibility to ABS issuers?

The WCP proposal gives us a good understanding of the input requirements but has little to say about output. We feel that in the interest of public accessibility, the SEC should specify the content and format of WCP output just as thoroughly as it will for WCP input. Leaving the nature of the output to the whim of the issuer would lead to a multiplicity of output interfaces that would hinder comprehension by investors. Equally importantly, it would prevent third parties from developing software to process the output of WCPs for further analysis and comparison.

The question of WCP testing also receives scant attention in the proposal. ABS issuers and their developers who will be responsible for writing WCPs are anxious to know how the SEC will assess a WCP for correctness. We understand that the SEC will not test WCPs before they are made available on EDGAR, but the SEC should develop a testing protocol that it can use in the event of a complaint to determine whether a given WCP accurately reflects the ABS for which it was submitted. This testing protocol should be published in advance so that WCP developers can know what standard to work toward.

We agree with the general idea of encapsulating the input data in XML, and feel that the same should be done for the output data. We request that all XML specifications be formulated in the meta-language of XML Schema Descriptions (XSDs) rather than the older Document Type Definitions (DTDs).

12. Should we require comments in the code that explain what each line does? Is this necessary given the narrative disclosure of the waterfall in the prospectus? If it is appropriate, are there any specific explanations we should require?

It would be unusual and confusing to have a comment for each line of code in a program. However, a certain amount of commenting is desirable. It would be sensible for the SEC to recommend that every function, class, and method begin with a brief descriptive comment. Many programming environments have a facility known as the documentation string, or doc string for short, that is intended for this purpose. Doc strings are built into the Python language in the form of the comment string, which is a string that optionally occurs as the first line of a function, class, or method.

Comment strings are not only useful to those who read the source code directly, but they also can be retrieved interactively through the Python interpreter or programmatically by a Python script. Encouraging WCP developers to write non-trivial comment strings would be a useful complement to the narrative disclosure in the prospectus.

14. Is our proposal to require credit card master trusts to report changes to the waterfall computer program on Form 8-K and file the updated waterfall computer program as an exhibit appropriate? Would the flow of funds, and thus the waterfall computer program, change over time? If so, how and why would it change? Should we require the waterfall computer program be filed at any other time? Should we require it be filed with each Form 10-D?

The SEC should require new filings whenever external events make the initial Waterfall program inaccurate.

15. Is the proposed requirement to provide the waterfall computer program with the proposed Rule 424(h) prospectus as of the date of filing and a final prospectus under Rule 424(b) as of the date of filing appropriate? Should the waterfall computer program be required to be filed at any other time? If so, please tell us why. As we discuss above in Section II.B.1.a., under our proposal, for material changes in information, other than offering price, which would include material changes to the waterfall computer program, a new Rule 424(h) filing would be required as well as a new five business-day waiting period.

Yes, it is appropriate to file the waterfall program at the date of filing when the final prospectus is due. It would be burdensome and unwieldy to mandate the filing of the waterfall program earlier, because CMO and ABS structures are in constant flux pre-closing. Until the underwriter is ready to close the deal, the waterfall may vary. Although buyers may *circle* a bond for purchase prior to close, at the close of the deal when the final waterfall program is submitted, the investors should then exercise their due-diligence duties and verify that their bonds indeed meet the criteria agreed upon earlier.

18. We propose to use existing submission types in order to enable filers to attach the asset data file as an exhibit. Tagging specifications that explain the requirements of the XML schema would be included in the proposed technical specifications. Are there other specifications that would be helpful that should be provided in the EDGAR Filer Manual for asset data files that are not currently included in other Technical Specifications? Please be specific in your response.

XML is a suitable format for asset data files and we support the use of XML schemas, specified either with the XSD language or the more specialized XBRL, to describe the required input structures. However, XML is not suitable for the submission of the waterfall computer program. We raise this point because the SEC proposal states, on page 23380 of the Federal Register (Vol. 75, No. 84), the following: "The waterfall computer program source code would be required to be submitted as tagged XML data."

One objection to using XML to submit source code is that a computer program is already structured in a very specific way defined by the programming language being used. Program statements, expressions, functions, comments, and so on must be formatted according to the syntax of the programming language. To attempt to redefine these structures in XML, or simply to surround the

whole program with a pair of XML tags, would add no value to the submission. In the best case, the user would have to strip the XML tags from the source code before running it. In the worst case, an XML processor might fatally corrupt the syntax of the computer program.

The use of XML to transmit a computer program is especially dangerous in the case of Python, which is a whitespace-sensitive language. Unlike most languages, which use braces or word tokens to delimit code blocks, Python requires the use of consistent indentation to show where a block begins and ends. This mandatory indentation makes Python programs easy to read on the part of humans, but it demands careful handling by text editing programs. Any process that tampers with the whitespace in a Python program risks making the program syntactically invalid or semantically incorrect.

The XML specifications promulgated by the World Wide Web Consortium (W3C) specify that an XML processor must preserve all content in an XML document without alteration, but in practice this is often not the case. There is an XML attribute named xml:space that may be used to declare the intention to preserve all whitespace in the enclosed content. If there is to be any situation where a piece of Python code is enclosed by XML tags, the xml:space attribute should certainly be declared with the *preserve* value. However, not all XML processors may implement this attribute correctly, and the end user would still have to strip the XML tags from the source code before executing it. Therefore, we recommend that the SEC avoid using XML altogether in the representation, storage, or transmission of Python source code.

Instead, we recommend that the SEC require that the waterfall computer program be submitted in the form of a text file, also known as a plain text or ASCII file. Text editors that are used for programming generally produce text files by default, so in most cases the source code will require no modification prior to submission. The only difficulty in sharing source code across computing platforms may be posed by variance in newline (also called end-of-line) representation. Some systems use the single ASCII character LF to mark the end of a line, while others use the two-character sequence denoted CR+LF. This variance usually causes no trouble because most text editors and program interpreters can handle both conventions. For the sake of maximal interoperability, however, it would be best if the SEC were to require the use of one or the other newline representation. We recommend the use of a single LF, in keeping with the predominant practice among system programmers.

19. Should we provide a transition period prior to the required compliance date that would allow filers to submit only test filings? Please be specific in your response.

A formal transition period would be a very good idea because the WCP requirement is without precedent. Although we anticipate that there will be widespread agreement around the notion that the input and output of WCPs should be rigorously specified for the sake of comparability and interoperability, there is likely to be much uncertainty about exactly what input and output specifications will work well in practice. The first attempt at producing a set of specifications will probably contain flaws that will only become evident once the market begins to use WCPs.

Therefore, we propose that the SEC provide for a staged transition in which the WCP requirement will not be mandatory and will be subject to revision at scheduled intervals in response to market feedback. Once the flaws in the WCP specifications have been ironed out and the transition period

has ended, all parties will have greater confidence in the feasibility and stability of the specifications. We request that the SEC schedule a transition period of at least twelve months, during which the WCP specifications would be subject to revision at the end of each quarter, and the final set of specifications would be expected to remain unchanged for a period of several years.

20. Is our proposal to permit the filing of an exhibit to disclose additional program functionality appropriate?

It would be best if the WCP did not, in fact, contain any additional functionality beyond what is required to give expression to the flow of funds. After all, the WCP is meant to act as a reference for those who seek to understand the precise workings of an ABS issue. Permitting the WCP to incorporate code that is not directly involved in the calculation of the flow of funds would open the door to increasingly bloated program submissions that defeat the original aims of clarity and transparency.

Code bloat is caused by the economics of software development: programs tend to incorporate software libraries (of which they employ a small part) and in turn are gradually incorporated into larger software libraries (of which they constitute a small part). Allowing supplementary functionality in the WCP would probably lead to large, convoluted submissions in which only a small proportion of the code is relevant to the ABS.

The only way to prevent code bloat in WCP submissions is by decree. The SEC should stipulate that a WCP submitted to EDGAR may only contain functions, methods, and data structures that are called upon at some point in the calculation of the output that is required from a WCP. If a WCP makes use of a library, it should be the developer's responsibility to extract from the library only those functions that are needed to calculate the required output. Similarly, if the WCP is part of a larger program that performs any calculation other than what is required, the developer should excise the superfluous code from the WCP. The resulting gains in brevity and simplicity would be very much to the investor's benefit.

If an issuer wished to supply additional program functionality beyond the WCP requirement, this should be done outside the framework of the WCP submission and perhaps outside EDGAR altogether. The WCP should stand as a clear and concise algorithmic description of the ABS issue in a consistent form that lends itself not only to ready comprehension by investors, but also to execution by third-party software applications that further analyze and compare the various market issues.

21. Are there any impediments that issuers would face if they are required to file the waterfall computer program on EDGAR?

The potential problem of someone hacking an EDGAR library that housed the final WCP exists, however unlikely; and the contents of a given WCP are not so unique as to entice theft. The concept is merely to make the WCP transparent and available to all investors.

In summary, we strongly support the SEC's current proposal to deal with this very important issue. We believe the proposal represents a significant step forward for rebuilding in a responsible manner these very important ABS markets. We welcome the opportunity to work with you and other industry players to help move the proposal forward. If there are any questions, please feel free to contact any of the

members of the SRR Working Group listed in the Appendix that follows.

Sincerely,

Joseph Masri
Chair, RMA Securitization Risk Roundtable
Member, RMA Market Risk Council

CC: William Githens, Chief Executive Officer, Risk Management Association
Aleem Gillani, Chair, Market Risk Council, Risk Management Association
Christopher Kunkle, Dir. of Securities Lending & Market Risk, Risk Management Association
Francis Garritt, Assistant Dir. of Securities Lending & Market Risk, Risk Management Association,

APPENDIX

Members of the Risk Management Association's Securitization Risk Roundtable participating in the Preparation and Review of this Response

Joseph Masri
Ariel Blumencwejg
Tania Fago
Franklin Robinson
Gabor Laszlo
Mike Laszlo
Michael Osinski
Olivier Lefevre
Richard Targett
Ashish Dev
Dan Rosen

Francis P. Garritt, Associate Director, **Risk Management Association**