Validating Time Efficiency of AOSR 2.0: A Novel WMS Planner Algorithm for SMEs, under Industry 4.0

Fareed Ud Din[1*], David Paul[2], Joe Ryan[1], Frans Henskens[1], Mark Wallis[1]

[1] The University of Newcastle, Callaghan, NSW, Australia.
[2] The University of New England, Armidale, NSW, Australia.

* Corresponding author. Tel.: +61 402743250; email: fareed.uddin@uon.edu.au
Manuscript submitted November 19, 2019; accepted December 16, 2019.
doi: 10.17706/jsw.15.2.53-61

Abstract: Agent Oriented Storage and Retrieval (AOSR) WMS planner algorithm is a part of the general Agent Oriented Smart Factory (AOSF) framework, which provides a comprehensive Supply Chain (SC) architecture to help bridge the gap between Industry 4.0 standards and SME-oriented setups. This paper provides validation of AOSR algorithm with respect to time efficiency on top of substantially improved performance efficiency in SME-oriented warehousing operations. This article, which is a part of a series of recent contributions, explains the efficiency of AOSR WMS planner algorithm in scenario-based test cases with experimentation in certain WMS Key Performance Indicators (KPIs).

Key words: Agent oriented smart factory (AOSF), small to medium size enterprises (SMEs), cyber-physical systems (CPS), agent-oriented storage and retrieval system (AOSR), warehouse management system (WMS)

1. Introduction

The inception of the concept of Industry 4.0 [1] is revolutionising recent industrial setups. In the fast-paced progression of Industry 4.0 implementation, Small to Medium Size Enterprises (SMEs) are not getting the expected benefits [2]. In order to help bridge this gap, the novel approach of the Agent Oriented Smart Factory (AOSF) framework presents a moderate-level semi-autonomous system for SMEs to apply a comprehensive SC framework under the umbrella of Industry 4.0 [3]. This article is part of an episodic series of a broad contribution of the AOSF framework and its associated Agent Oriented Storage and Retrieval (AOSR) Warehouse Management System (WMS) strategy, which includes a general high-level AOSF framework [3]; its extended visualisation as a Cyber Physical System (CPS) [4]; problem and domain definitions to build the baseline model for AOSR [5]; *AOSR's 6-Feature Strategy* and general work-flow [6]; and a thorough performance validation of AOSR in comparison with standard WMS strategies [7] including multiple warehousing and product placement/retrieval mechanisms, e.g. Zoning Logic, FIFO Logic and Pick from/Put to the Fewest Logic [8].

This article includes validation test cases and scenarios applied to the AOSF recommended WMS strategy, AOSR 2.0, to affirm the validity of the overall system with respect to time efficiency. It includes test scenarios within the supply chain of a firm and relates it with different possible cases of information exchange from the front-end customer side (CRM) and back-end supplier side (SCM). Detailed results from prototyping the hybrid-logic-based AOSR algorithm are included, which combines not only all the aforementioned logic schemes but also the `Pick from/Put to the Nearest logic', in order to reduce the overall activity-time within the shop-floor. This article also includes discussion about how the *6-Feature*

Strategy recommended by the AOSR system helps in bringing improvement and pro-activeness within a warehouse.

2. Discussing Time Efficiency of AOSR 2.0

AOSR algorithm provides a simple but overarching solution to the problems of scheduling products and their slotting and re-slotting within an SME-oriented warehouse. It focuses on the main reasons that cause major problems in warehouse management by, for example, attempting to reduce the number of products in receiving areas (RAs) and expedition areas (EAs) and maximising the number of products within the defined racks [8]. Based on percepts from the environment and descriptive initial states of the system, AOSR generates a comprehensive placement plan, utilising predefined sets of actions. The proactive nature of AOSR strategy provides help in managing the space within the warehouse to cater to upcoming products and its hybrid strategy supports the efficient slotting and re-slotting of products between different locations within the shop-floor. In our previous work [6] we discussed the 6-Feature Strategy and hybrid logic selection as proposed by AOSR heuristics [7].

AOSR algorithm is based on the classical BDI agent model structure [9] and follows the constructs and agent classification of AOSF framework [3], which shapes it into a dynamic solution in order to support operational flexibilities in the future. AOSR Planner Agent (PA) utilises Information-Sets (as discussed in our other work [10]) related to different classification of products, racks, and their characteristics, which serve as the Belief Base for PA, while information related to current stock levels and system states serves as the Knowledge Base for it, which may be further updated by actuators. For PA, three main segments play the role of actuators: (i) Placement Generator, (ii) Extract Placement, (iii) Search Rack. These actuators sense the percepts coming from the environment and update the Belief Builder and Knowledge Builder in the general architecture of the AOSR system (details about all these segments are included in our previous work, discussing the heuristics of AOSR 2.0 [7]).

All the heuristics of AOSR 2.0 have been implemented in JADE [11], which provides simplicity with the flexibility to design multiple agents along with the facility of sniffer agent interfaces to monitor the overall agents' activity. Constraint-based tests have been applied to acquire results by applying AOSR 2.0 strategy in contrast to the standard WMS approach (discussed in detail in our previous work [7]) to see if the issues in warehouse management can be reduced by employing a moderate level semi-autonomous AOSR solution.

The prototype to validate AOSR strategy utilizes comprehensive test datasets extracted from online sources provided by DGI Global [12] and Eurosped [13] warehousing and logistics companies as detailed in our other work [10]. This comprehensive data set consists of the information of several characteristics of products e.g. SKUs, quantity and products classes, from several different industrial sectors e.g., electronics industry, medical industry, textile firms, paint and glass industry. The 6-Feature Strategy of AOSR [6] attempts to provide a solution to SMEs' problems in warehousing, such as wandering items/picking lists [14], [15], inaccurate stock value at runtime [16], unmanaged receiving and expedition areas [17], unmanaged storage capacity [18] and inappropriate retrieval scheduling [19].

While validating the technical solutions, execution time is always an important factor to be considered. In order to evaluate the time efficiency of AOSR strategy, several test cases are applied. These test cases and scenarios are categorised as below:

- Gradually Reducing Search Space;
- Gradual Change in Product Characteristics; and
- Random Cases.

For all these scenarios, the results are acquired from both sides of the general AOSF recommended SC architecture: Enterprise Central Unit (ECU) side and Customer Relationship Management (CRM) side. These

results with their implications are discussed in the following Sections.

2.1. Scenario of Gradually Reducing Search Space

In literature, several other researchers have computed the standard computation times on Pentium series computers for similar heuristics-based scenarios but in a different context of location-allocation for vehicle-routing problems, and the average CPU time lies between 0.52s to 8.57s [20], [21] or 0.62s to 10.21s [22] with different approaches. Research suggests that, on average, less than or equal to one second (=< 1s) is considered a standard CPU processing time for inbound logistics [23], regardless of hardware configuration.

AOSR maintains a balanced execution time to produce results as shown in Fig. 1; all of the transactions, performed on an Intel (R) Core (TM) i5 computer, having 3.7 GHz clock rate and 64-bit Mac Operating System (OS), took less than 0.02s, which reflects its efficiency with respect to time as well. This execution time is comparable with the other standard approaches validated and tested by Waris *et al*. [18], on a similar hardware configuration (Intel Core i5 with 64-bit OS), where the average execution time for parsing information, in a similar scenario, is 0.021s (This article does not include explicit test cases for validating the execution time for other existing approaches).

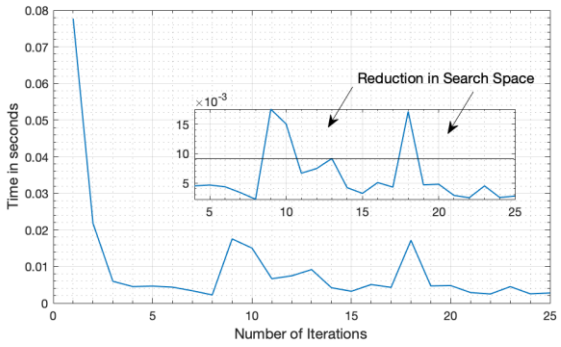


Fig. 1. Efficiency of ECU side of AOSR algorithm with gradual reducing search space.

A closer look at Fig. 1 can explain that, with the reduction in search space, it takes less time to compute and generate results. The ECU component utilises *Percept-Builder* (as detailed in heuristics of AOSR 2.0 [7]), which employs caching techniques for finding an appropriate rack for the upcoming products. AOSR algorithm takes less time when it is in the same iteration and when it switches the iteration the time taken increases abruptly (but not more than 0.02s) and then again reduces gradually in the same iteration. This phenomenon can be observed at iterations 9 and 18.

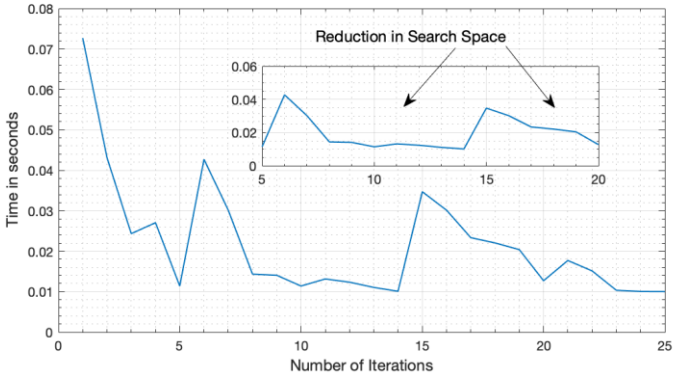


Fig. 2. Efficiency of CRM side of AOSR algorithm with gradual reducing search space.

Similarly, the CRM side of AOSR algorithm also takes the same strategy and almost the same trend in execution time as reflected in Fig. 2. The time taken to execute CRM side transactions is a bit higher than the transaction time on the ECU side as it utilises a double iterative strategy, in order to update the inventory as well as the capacity in the stock. Even after performing almost double the number of tasks as its partner side, only one iteration took over 0.04s (in iteration 6) with most of them taking around 0.02s. Also, the reduction in search space reduces the execution time as well, which can be seen between iteration 5 to 13, 14 to 22 and 23 to 30.

2.2. Scenario of Gradual Change in Product Characteristics

The test cases to validate the AOSR strategy include 20 different classifications of products as detailed in our other work [10], so a gradual change in characteristics results in a gradual decrease in execution-time.

The results taken after applying the test case with a gradual change in product characteristics are represented in Figure [3]. *Percept Builder()* attempts to find the local optimal for every product characteristic and, in case of a change in characteristics, it exits the loop and attempts to build a new cache and starts searching for the optimal value again.

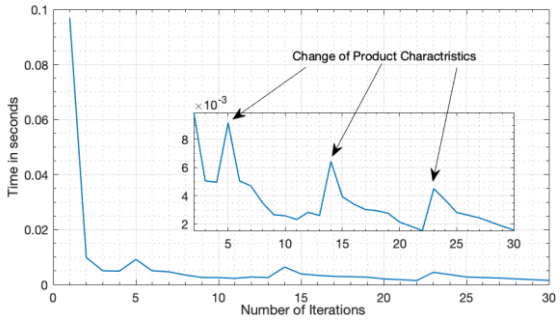


Fig. 3. Efficiency of ECU side of AOSR algorithm with gradual change in characteristics.

As reflected in Fig. 3, all the iterations take less than 0.01s which is a great execution time for an algorithm like AOSR, which interacts with the environment and computes the plan for the whole warehouse shop-floor. A closer look can explain that even at iterations 5, 14 and 23, when the characteristics change, it still took less than 0.01s and all the corresponding iterations were completed within about 0.005s.

Similarly, while executing the test case on the CRM side, as represented in Fig. 4, the gradual change in characteristics reduces the execution time but leaving an iteration and initiating new caching memory takes a bit more time because of its double iterative strategy. Even when building a new memory base, even the first iteration takes less than 0.06s, with all other iterations where the characteristics change having an execution time around 0.03s. All the other iterations take less than 0.02s, which demonstrates the consistency of the overall AOSR strategy.

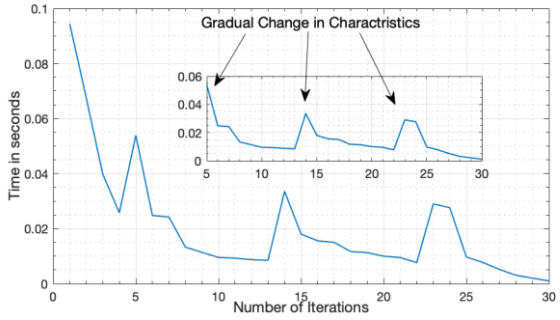


Fig. 4. Efficiency of CRM side of AOSR algorithm with gradual change in characteristics.

2.3. Scenario of Random Cases

For a complete validation, a set of 25 random test cases (with non-sequential product characteristics) is applied to AOSR and performance is seen to be consistent. These test cases include the random data from an already existing industrial data set (as used in cases mentioned above) to ensure the non-sequentiality of iterations. The extracted results for the ECU side are shown in Fig. 5, where the maximum time taken to extract the right information and activate the actuator is 0.022s. This demonstrates the efficiency and consistency of AOSR, even in random test cases. The least time taken is 0.006s, when the system must find and allocate space to a product where there is no product already stored, so the process remains quite simple and quick. On average all transactions took about 0.013s to compute the comprehensive product placement plan.

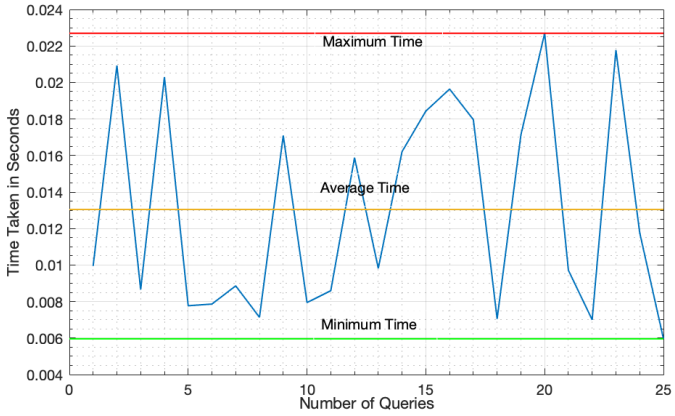


Fig. 5. Time taken AOSR with random test cases.

Similarly, the results extracted from the CRM side by applying the random test cases are reflected in Fig. 6. In the case of unavailability of space for a certain product, the AOSR utilises its re-slotting strategy where computation is then performed three times (to pool the information from Advance Shipment and Delivery Notice (ASN/ADN), re-slotting and then slotting the products if needed) by defined algorithmic heuristics to check and manage if there is a need to re-slot the products.

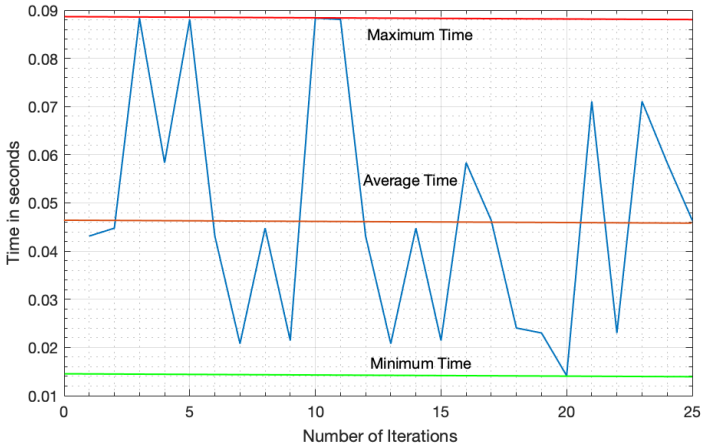


Fig. 6. Time taken AOSR with random test cases on CRM side.

In order to manage the space with efficiency, the CRM side of AOSR algorithm takes more time than the ECU side because it includes searching and updating both the racks and the inventory level. The maximum

time taken by the CRM side of AOSR was 0.09s, which is not a very high computation time for the algorithm to build, extract and perform transactional and analytical information, as presented by the study conducted in [21]. On the CRM side, when the required product is in the nearest rack with available matching quantity, the time taken is less than 0.02s as shown at iteration 20 in the graph. On average, the CRM side of AOSR took 0.047s to perform the task to satisfy requirements. The computation time taken by AOSR strategy falls well inside the time limits (between 0.52s to 8.57s [19] or 0.62s to 10.21s [20]) tested by several different studies such as presented in [19], [20], [21]. Thus, the efficiency and hybrid approach of AOSR makes it suitable for the industry where agility and customisation are the main metrics of success.

3. Conclusion

In this article, we have discussed results from different types of test cases to thoroughly validate the AOSR Planner Algorithm with respect to time efficiency. The test cases analysed the products stored in racks, in EA and in RA, each with respect to two different system sides: Enterprise Central Unit (ECU) and Customer Relationship Management (CRM) side. The successful and positive results, from all the scenarios and test cases, highlight the overall performance efficiency of AOSR algorithm in association with its parent AOSF framework. In future, the AOSF/AOSR strategy will be subjected to an incremental development in other open areas of supply chain e.g. Plant Maintenance, Procurement Automation and Product Requirement Scheduling. Also, other state of the art technologies e.g. Big Data and IoT can be armed with this framework to provide more robust features. Improvisation on the cloud side could also be an interesting future work in this regard. For WMS strategy some other features can be incorporated, such as the movement of forklifts within the shop-floor or to train the placement generator algorithm to be self-configured based on past event and historic data.

Conflict of Interest

The authors declare no conflict of interest.

Author Contributions

Fareed Ud Din: Conducted the primary research and drafted the article.
David Paul: Provided active feedback on the drafts for revision.
Joe Ryan: Supervised the workflow of the research.
Frans Henskens: Supported and overlooked the research idea overall.
Mark Wallis: Analysed the data and initial experiments.
All authors had approved the final version.

References

[1] Industry in Germany, German Trade and Industry (GTAI). (2011). Retrieved from: https://www.gtai.de/GTAI/Navigation/EN/Invest/industrie-4-0.html

[2] Muller, J. M., & Buliga, K. I. V. O. (2018). Fortune favors the prepared: How SMEs approach business model innovations in Industry 4.0. *Technological Forecasting and Social Change 132*.

[3] Din, F. U., Henskens, F., Paul, D., & Wallis, M. (2018). Agent-oriented smart factory (AOSF): An MAS based framework for SMEs under Industry 4.0. *Proceedings of the KES 190 International Symposium on Agent and Multi-Agent Systems: Technologies and Applications* (pp. 44-54).

[4] Din, F. U., Henskens, F., Paul, D., & Wallis, M. (2019). Extended agent-oriented smart factory(xAOSF) framework as a conceptualised CPS with associated AOSR-WMS system.

[5] Din, F. U., Henskens, F., Paul, D., & Wallis, M. (2018). Formalisation of problem and domain definition for

agent oriented smart factory (AOSF). *Proceedings of the IEEE Region Ten Symposium*, 265-270.

[6] Din, F. U., Henskens, F., Paul, D., Wallis, M., & Hashmi, M. A. (2019). AOSR-WMS planner associated with AOSF framework for SMEs.

[7] Din, F. U., Henskens, F., Paul, Ryan, J. F., & Henskens, M. W. (2019). AOSR 2.0: A novel approach and thorough validation of agent oriented storage and retrieval WMS planner for SMEs.

[8] Golovatova, A., & Jinshan, Z. (2010). Optimization of goods incoming process. Master's thesis, University of Boras, Sweden, Sweden.

[9] Russell, S., Norvig, P., & Intelligence, A. (1995). *A Modern Approach*. Artificial Intelligence, Prentice-Hall, Englewood Cliffs 25 - 27.

[10] Din, F. U., Ryan, J., Henskens, F., Paul, D., & Wallis, M. (2019). Validating agent oriented smart factory (AOSF) in comparison with linear supply Chain.

[11] Java agent development framework. (2017). JADE open source project: Java agent development environment framework. Retrieved from: http://jade.tilab.com/

[12] Global, D. G. I. (2018). Warehouse prducts classes. Retrieved from: http://www.dgiglobal.com/classes

[13] EuroSped, (2018). Dataset Information for warehouse and logistics. Retrieved July 25, 2018, from http://www.eurosped.bg/en/eurolog-warehouse-logistics-4pl/

[14] Gu, J., Goetschalckx, M., & McGinnis, L. F. (2007). Research on warehouse operation: A comprehensive review. *European Journal of Operational Research*, *177(1)*, 1-21.

[15] Business2Community, issues in warehouse management systems. (2018). Retrieved July 17, 2018, from https://www.business2community.com/product-management/top-5-warehouse-management-problems-solve-02027463

[16] Poon, T., Choy, K., Chow, H. K., Lau, H. C., Chan, F. T. & Ho, K. (2009). A RFID case-based logistics resource management system for managing order-picking operations in warehouses. *Expert Systems with Applications*, *36(4)*, 8277-8301.

[17] Richards, G. (2017). Warehouse management: A complete guide to improving efficiency and minimizing costs in the modern warehouse. Kogan Page Publishers.

[18] Lu, W., Giannikas, V., McFarlane, D., & Hyde, J. (2014). The role of distributed intelligence in warehouse management systems. *Annual Reviews in Control*, 63-77.

[19] Li, L. (2007). Supply chain management: Concepts, techniques and practices: Enhancing value through collaboration, World Scientific Publishing Co Inc.

[20] Waris, M. M., Sanin, C., & Szczerbicki, E. (2018). Smart innovation engineering: Toward intelligent industries of the future. *Cybernetics and Systems*, *49(56),* 339-354.

[21] Wang, X., Sun, X., & Fang, Y. (2005). A two-phase hybrid heuristic search approach to the location-routing problem. *Proceedings of the International Conference on Systems, Man and Cybernetics*.

[22] Wu, T. H., Low, C., & Bai, J. W. (2002). Heuristic solutions to multi-depot location routing problems. *Computers & Operations Research*, *29(10)*, 1393-1415.

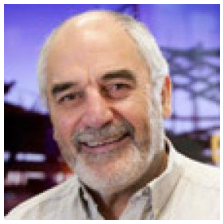
[23] Schoneberg, T., Koberstein, A., & Suhl, L. (2010). An optimization model for automated selection of economic and ecologic delivery profiles in area forwarding based inbound logistics networks. *Flexible Services and Manufacturing Journal*, *22 (34)*, 214-235.

Fareed Ud Din is a researcher in distributed computing and a recipient of several academic performance awards, including Best Research Paper Award, Best Technical Presenter Award and Best Academic of the Year Award. He has authored a total of 15 peer-reviewed publications overall with the submission of his PhD thesis at the University

of Newcastle, NSW, Australia. He has earned no less than 6 years uninterrupted teaching and research career over in Pakistan and Australia. He has been a part of several inter-disciplinary research groups e.g. Information Systems Group and Smart Grid Group at the Superior University, Lahore, IoT Research Group Information Technology University, Lahore and Distributed Computing Research Group, The University of Newcastle, NSW. His research interests include distributed computing, multi-agent systems, information systems, cyber physical systems and Industry 4.0.

David Paul is a computer scientist with a research focus in service-oriented computing, the Internet and distributed systems, security and privacy, and teaching and education. In the past, he has been working on nation-wide research projects such as the Australian Schizophrenia Research Bank and the QuON Web survey system. He received his PhD from the University of Newcastle (UoN), Australia in 2012 and continued to work as a member of the Australian Schizophrenia Research Bank till he joined the University of New England (UNE) in 2015 as part of the new interdisciplinary computational science team. At UNE his research interests have broadened into areas such as agriculture and sports science such as the projects: SheepCRC, ASKBILL and RamSelect systems, which primarily focuses to support farmers to profile their livestock and manage the farms more efficiently.

Joe Ryan holds an extensive teaching and research experience since 1990 when he joined the University of New England (UNE), Australia and later The University of Newcastle (UoN), Australia in 1992. He has been with the University of Newcastle since then except for a four period (2004 - 2008) when he served as a lecturer at University of Ballarat. He has produced over a hundred research publications under his name, including two books. His research expertise includes several diverse areas with a focus on Graph Theory and he has a vast network of collaboration around the globe, including Indonesia, Czech Republic, Slovakia, Spain, UK, USA, Canada, Germany, France, India and, of course, Australia.



Frans Henskens started his academic career at the age of 40 and has had very extensive experience in inter-disciplinary research collaboration for an uninterrupted more than 25 year period, starting at the University of Sydney in 1995 and later with research groups in UoN (Mental Health) and QUT (Bioinformatics) in 1998. Frans has produced more than 200 peer-reviewed research publications and supervised close to 20 research theses. His research interests include a wide variety of recent topics, including engineering of flexible software systems, bioinformatics, operating systems and computer forensics, distributed and grid computing, resilience and availability in database systems, use of persistent stores for bulk data storage and manipulation, and use of (particularly mobile) computing to investigate and influence health behaviour.

Prof Henskens has received several research and teaching awards, namely, the University of Newcastle Vice Chancellor's Award for Teaching Excellence, the Australian College of Educators NSW Quality Teaching Award and the Vice Chancellors Award for Research Supervision Excellence. He has also held several university headships positions, including Associate Dean, Head of School and Director of Distributed Computing Research Group.

Mark Wallis is a software and network engineer who specialises in software architecture, secure networks, L4+ networking, auditing, policy, project management and education. With years of experience in the field, Mark works with his colleagues to design and implement secure software, network and integration solutions. Mark is a Conjoint Lecturer at the University of Newcastle and has completed his PhD in distributed component-based systems. His teaching interests include Enterprise Software Architecture, Network and Distributed Computing, Distributed Operating Systems, Computing Fundamentals, Introduction to Professional Engineering and Web Engineering.